Universidade de Aveiro
2012

Departamento de
Eletrónica, Telecomunicações e Informática

Pedro Filipe Seabra da Costa

Algoritmos para recetores coerentes em redes óticas de acesso

Universidade de Aveiro
2012

Departamento de
Eletrónica, Telecomunicações e Informática

Pedro Filipe Seabra da Costa

Algoritmos para recetores coerentes em redes óticas de acesso

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações, realizada sob a orientação científica do Dr. Paulo Miguel Nepomuceno Pereira Monteiro, Professor Associado do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, com co-orientação do Prof. Rui Sousa Ribeiro, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

Apoio financeiro da FCT e do
FSE no âmbito do III Quadro
Comunitário de Apoio.

o júri

presidente

Prof. Dr. Nuno Miguel Goncalves Borges de Carvalho
Professor associado com agregação da Universidade de Aveiro

Prof. Dr. Henrique José Almeida da Silva
Professor associado da Universidade de Coimbra - Faculdade de Ciências e Tecnologia

Prof. Dr. Paulo Miguel Nepomuceno Pereira Monteiro
Professor associado da Universidade de Aveiro (orientador)

Prof. Dr. Rui Fernando Gomes de Sousa Ribeiro
Professor auxiliar da Universidade de Aveiro (co-orientador)

agradecimentos

Em primeiro lugar, agradeço e dedico este trabalho à minha família, em particular aos meus pais e ao meu irmão, pelo constante auxílio demonstrado ao longo da minha formação, tanto pessoal como académica, e sem os quais este trabalho não teria sido possível.

Agradeço ao Professor Paulo Monteiro, ao Professor Rui Ribeiro e ao Doutor Miguel Drummond pela disponibilidade e orientação à elaboração desta dissertação. Agradeço também ao Professor Rogério Nogueira pela aquisição da FPGA, e ao Professor Arnaldo Oliveira, pelo auxílio prestado na área dos sistemas reconfiguráveis.

Deixo também um agradecimento ao Engenheiro Nelson Ribeiro pelas ocasionais trocas de ideias que provaram ser uma mais-valia à realização deste trabalho.

Por fim, agradeço aos colegas pelo companheirismo e apoio demonstrado ao longo destes cinco anos. Em particular, um obrigado ao colega Malafaia, pela constante palavra amiga e encorajadora, e ao colega Bruno, pelas constantes discussões e dúvidas esclarecidas na realização deste trabalho.

palavras-chave Comunicações óticas, deteção coerente, redes de acesso, pós-compensação digital, dispersão dos modos de polarização, FPGA

resumo O constante aumento na necessidade de largura de banda tem levado ao desenvolvimento de redes WDM com elevada densidade de canais, assim como à adoção de formatos de modulação avançados de elevada eficiência espectral. Tais formatos requerem a substituição dos atuais recetores de intensidade por sistemas de receção coerente, onde toda a informação presente no domínio ótico é adquirida. Uma vez que toda a informação é adquirida, torna-se possível e vantajoso efetuar pós-compensação digital das distorções introduzidas na transmissão em fibra ótica.

Nesta dissertação é estudada a pós-compensação digital dos efeitos da dispersão dos modos de polarização numa rede WDM, onde cada canal é modulado numa única polarização. Através de simulação em MATLAB foram desenvolvidos e testados diversos algoritmos que permitem realizar a tarefa em questão. Os mesmos algoritmos foram depois implementados em hardware, com recurso a uma FPGA. Foi desenvolvida uma interface de teste em MATLAB que, em comunicação com o hardware, permite testar a performance dos diversos algoritmos. Por fim, foi escolhida a melhor opção para implementar um equalizador para a dispersão nos modos de polarização, tendo em conta o desempenho e requisitos em termos de hardware de cada implementação.

keywords optical communications, coherent detection, access networks, digital post compensation, polarization mode dispersion, FPGA

abstract The constant demand for bandwidth has led to the development of ultra-dense WDM networks as well as advanced modulation formats with high spectral efficiencies. Such modulation formats require coherent receivers, which enable acquiring all the information present in the optical field. Since all the information is received, effective digital post-compensation of transmission impairments becomes possible.

The subject of this dissertation is digital post compensation of polarization mode dispersion in a WDM network comprising single-polarization channels. Several algorithms were developed and tested in a MATLAB environment to perform the task in hands. Such algorithms were then implemented in hardware, resorting to a FPGA. A MATLAB interface between the computer and the hardware was developed, enabling the assessment of the performance of the algorithms. Finally, the best algorithm was chosen by comparing the performance and hardware requirements for each implementation.

Content

List of acronyms

ADC	Analog to digital converter
ASIC	Application specific integrated circuit
ASK	Amplitude shift keying
AWGN	Additive white Gaussian noise
BD	Balanced photodetector
BER	Bit error rate
BPSK	Binary PSK
CD	Chromatic dispersion
CLB	Configurable logic block
CMA	Constant modulus algorithm
DAC	Digital to analog converter
DCF	Dispersion compensating fiber
DD	Direct detection
DD-LMS	Decision-driven least means square
DGD	Differential group delay
DOP	Degree of parallelization
DQPSK	Differential QPSK
DSK	Dispersion shifter fiber
DSP	Digital signal processing
EAM	Electro-absorption modulator
EDFA	Erbium doped fiber amplifier
EOM	Electro-optic modulator
EPLL	Electrical PLL
EPON	Ethernet PON
EVM	Error vector magnitude
FEC	Forward error correction
FF	Flip-flop
FTTX	Fiber to the X
FIFO	First in first out memory
FIR	Finite impulse response filter
FPGA	Field programmable gate array
FSK	Frequency shift keying
FWM	Four wave mixing
GPON	Gigabit PON
HDL	Hardware description language
IF	Intermediate frequency
IIR	Infinite impulse response filter
IM	Intensity modulator
IPCore	Intellectual property core
IQ	In-phase and quadrature modulator
IS	Impulse shaper
LO	Local oscillator
LPF	Low pass filter
LUT	Look up table
MMF	Multi mode fiber
MZM	Mach-Zehnder modulator
MC	Monte Carlo
NGOA	Next generation optical access
NRZ	No return to zero

ODN	Optical distribution network
OLT	Optical line terminal
ONU	Optical network unit
OOK	On-off keying
OPLL	Optic PLL
OTG	Optical transmission group
PBS	Polarization beam splitter
PDM	Polarization division multiplexing
PLL	Phase locked loop
PM	Phase modulator
PMD	Polarization mode dispersion
PON	Passive optical network
PSK	Phase shift keying
QAM	Quadrature amplitude modulation
QPSK	Quaternary PSK
RZ	Return to zero
SMF	Single mode fiber
SNR	Signal to noise ratio
SOP	State of polarization
SPM	Self phase modulation
SSMF	Standard SMF
TDM	Time division multiplexing
VCO	Voltage controlled oscillator
WDWDM	Ultra dense WDM
WDM	Wavelength division multiplexing
XPM	Cross phase modulation

List of symbols

∂	Angle used in the rotation matrix
B	Birefringence matrix
c	Speed of light
D	Total chromatic dispersion
D_i	Used to specify and explain a delay introduced in a previous figure
E	Total amplitude of the electrical field of an optical signal
$\hat{e}$	Unitary vector indicating the polarization of an optical signal
E_1, E_2	Input electrical fields at the balanced photodetector
E_{in}	Electrical field at the input of an optical modulator
E_{lo}	Electrical field for the optical local oscillator
E_{os}	Electrical field of a modulated optical signal
E_{out}	Electrical field at the output of an optical modulator
E_{ox}, E_{oy}	Electrical field of the optical signal for each polarization
E_s	Digital electrical signal containing phase information
E_T	Digital electrical signal after compensation considering noise
E_x, E_y	Digital electrical signal received for each polarization
E_x^1	Digital signal in polarization x after phase compensation
g	Gain (attenuation) in the total received power
g_{agc}	Gain introduced by the automatic gain control to compensate for g
I	In-phase component
I_{out}	Output current for a balanced photodetector
k	Symbol identifier
L	Travelled distance inside an optical fiber
m	Channel indicator in a WDM network
n	Identifier for an optical fiber segment
N_{bits}	Number of bits per symbol
N_{sym}	Number of symbols in a constellation
N_x, N_y	Additive white Gaussian noise introduced in each polarization
N_T	Noise present in the digital signal after compensation
P_e	Power of the received optical signal
P_{in}	Power introduced to the fiber
P_{lo}	Power of the local oscillator
P_N	Noise power
P_{ox}, P_{oy}	Power of the optical signal for each polarization
P_s	Power of an optical signal
Q	Quadrature component
r	Responsivity for a balanced photodetector
R	Rotation matrix
S	Symbolic representation
S_i	Used to specify and explain a signal introduced in a previous figure
T	Jones matrix
u_1, u_2	Coefficients used in the Jones matrix
V	Voltage used in a phase modulator
V_1, V_2	Voltages used in a Mach-Zehnder modulator
V_π, V_{pi}	Voltage required by a modulator to invert the electrical field of an optical signal
w	Angular frequency
α	Fiber attenuation

β	Generic phase constant for an optical signal
β_f, β_s	Phase constant for the fast and slow polarizations
δ	Phase difference between polarizations
Δ	Intermediate frequency at an heterodyne receiver
$\Delta\tau$	Differential group delay
ε	Factor used in the dimensioning of infinite impulse response filters
η_1, η_2	Refractive index for the core and cladding
η_{eff}	Effective refractive index
η_f, η_s	Refractive index for the fast and slow polarizations
θ	Phase rotation introduced in the birefringence matrix
θ_s	Phase modulation present in a signal
λ	Wavelength
ρ	Power split ratio between polarizations (indicates the power in the polarization x)
ϕ	Phase mismatch between the local oscillator and the received signal
φ	Phase information for a symbol

List of figures

List of tables

1. Introduction

1.1. Optical communication systems

The first optical systems were investigated in the 1970s. The low attenuation of optical fiber meant that larger repeater spacing was allowed when compared with the copper technology, leading to lower implementation costs. These optical systems were based in intensity modulation (IM) at the transmitter and direct detection (DD) at the receiver, using a single channel. Due to their simplicity and low cost, IMDD was used in most optical communication networks. Earlier optical systems operated at $0.85\mu m$. However, due to the limitation to the data rate imposed by chromatic dispersion, systems moved to $1.3\mu m$, where chromatic dispersion presents its minimum. Experiments in the mid 1970s showed that, for $1.5\mu m$, fiber presented the minimum attenuation. Today optical systems operate at this wavelength region [1].

From multimode fiber (MMF), where the bit rate was limited to 100Mb/s due to multimode dispersion, systems evolved to single mode fiber (SMF) and rates of 2Gb/s were achieved in the 1980s. By this time, being the most cost-effective method to transmit telecommunications traffic, optical networks were adopted by the telecom operators [2]. In addition, with the invention of the erbium-doped fiber amplifiers (EDFA), the first wavelength division multiplexing (WDM) experiments were performed. This technology quickly increased the capacity of optical fibers by transmitting more and more channels simultaneously. Using WDM technique, the fiber capacity has increased by over 10.000 times in 20 years [1]. In fact, in 2008 a transmission of 25.6Tb/s over optical fiber was demonstrated, using a WDM network [3].

With advances in laser manufacturing, new lasers with reduced phase noise made coherent reception a viable option [4]. Instead of detecting only the power present in the fiber, coherent receivers allow the acquisition of amplitude and phase information. With these receivers, advanced modulation formats became more attractive and a reality. While in IM only the amplitude of the electrical field of the optical laser could be modulated, advanced formats can operate on the amplitude, phase and polarization of the optical signal, in a symbolic representation of the transmitted data. By increasing the number of bits represented by each symbol, high data rates can be transmitted over lower symbol rates, resulting in smaller bandwidths, which advantageously reduce the effects of fiber propagation impairments and increase the spectral efficiency. Coherent systems are considered the future-proof technology to implement high speed long-haul transmission systems [5].

1.2. Optical access networks

The optical networks capabilities over other telecommunications technologies led to their generalized implementation by the telecom operators, not only in the core networks, where high capacity is fundamental, but also in the access networks, in a fiber-to-the-x

(FTTX) architecture. Most of the access networks currently deployed are time division multiplexing (TDM) passive optical networks (PON) [6]. These networks employ GPON and EPON technologies, where the medium is time-shared between all clients.

The increasing number of broadband users and the bandwidth required by new video-centered services are pushing TDM networks to their limits and forcing the telecom operators towards new solutions for the access networks [7]. Ultimately, the use of WDM combined with advanced modulation formats is the only technology able to cope with the increasing demands of bandwidth, supporting a higher number of clients and higher bandwidth per client in a single fiber. Individual wavelengths are assigned to different clients, increasing the privacy / security and reducing the bit rate of operation for each client [6].

1.3. Signal equalization and digital signal processing

In the first optical communication systems the compensation of fiber impairments was performed in the optical domain, using dispersion compensation fibers (DCF) or dispersion shifted fibers (DSF), to compensate chromatic dispersion (CD) [2], and using polarization controllers (PCs) and variable delay lines to compensate for polarization mode dispersion (PMD) and differential group delay (DGD) [8]. These optical compensation techniques are transparent to the data rate and modulation format of the carried signal, being suitable to systems with high data rates [9]. However, this type of compensation is expensive due to the cost of optical discrete components.

The use of electronic equalization in optical systems began with the pre-compensation of chromatic dispersion in IMDD systems through the introduction of chirp at the transmitter [10]. By predicting the channel effects, the pre-compensation techniques introduce a distortion in the signal. While traveling along the fiber, the emitted pulses regain their shape, arriving at the receiver fully recovered. The other option is called post compensation, where the signal is analyzed and corrected at the receiver, and was first used in the 1990s [11] for DD systems. When DD is used the phase information encoded in the optical signal is lost, and zero-penalty equalization cannot be achieved. If, however, coherent reception is used, all the information present in the electrical field of the input optical signal is received, enabling complete equalization of the received signal [12].

Since all the information contained in the received light can be received in the electrical domain, DSP can be used to perform signal equalization and retrieve the transmitted information. New advances in the processing speed and power consumption of digital equipment have made digital equalization a viable option [2]. These result in cheaper systems, where adaptive algorithms can be used to compensate time-varying transmission impairments [9].The DSP algorithms are mainly used to perform carrier phase error correction [13] and compensate fiber propagation impairments, from linear [14] to non-linear effects [15].

The use of forward error correction (FEC) codes is fundamental in optical communication systems since, by adding an overhead of 7%, it becomes possible to obtain error-free transmission for much lower SNRs. [16].

1.4. Motivation

Due to the increase of bandwidth demand, coherent systems are becoming progressively more common, as advanced modulation formats allow higher spectral efficiencies. Since the optical signal is entirely received to the digital domain, DSP can be used to equalize the received signal, implemented using ASICs or FPGAs. Here the major limitation is the processing speed. However, new advances have produced FPGAs with processing rates up to 10Gb/s [2], enabling their use in optical coherent systems. Different algorithms have been proposed for digital equalization. Particularly considering equalization of polarization mode dispersion (PMD), there are different algorithms that allow, in polarization division multiplexed (PDM) systems, the separation of the two polarizations [14, 17, 18]. In systems with single polarization such algorithms can be simplified. However, they are only able to track slow variations of the state of polarization of the received signal. A simple algorithm has also been suggested [19] to perform equalization specifically when single polarization is used. However, when aiming to the hardware implementation, this solution presents some disadvantages.

1.5. Objectives and dissertation structure

The main objective of this dissertation is the study of post-compensation algorithms for PMD equalization when single polarization is transmitted. This study focuses the implementation of such systems in the next generation optical access (NGOA) network. Starting with a simplification from the algorithms used in polarization demultiplexing [14], followed by a possible solution to the problem [19], new algorithms and different approaches are studied, aiming to an increase in the efficiency and a reduction in the hardware requirements. The first steps consist in a behavioral simulation, where the different algorithms are developed and tested using MATLAB and SIMULINK. In order to obtain realistic test samples, where the different fiber and transmission impairments are considered, a conceptual optical network was implemented using OSIP. The different algorithms were also implemented in a FPGA, to perform tests under real hardware limitations.

The present document is divided in nine chapters. In the current chapter a short introduction to optical communication systems, optical access networks and signal equalization techniques is presented. In the second chapter optical coherent systems are studied, considering the different modulation formats and implementations for the transmitter and receiver. Fiber impairments are also studied, in particular PMD, the main subject of this dissertation. In the third chapter the features and constituents of NGOA are presented. This will be the scenario under which simulations and circuits will be designed. In the fourth chapter the different development environments are presented, taking into account the blocks developed in SIMULINK, the simulation of an optical network through OSIP and the setup used in the hardware tests, where MATLAB communicates with the FPGA. In the fifth chapter the different algorithms are introduced and simulations performed to test and compare their behaviors. These same circuits will then be implemented and tested in a FPGA, as described in chapter six. After analyzing the different algorithms, combining the performance results and the hardware requirements,

the best solution is chosen and used to implement the desired algorithm in chapter seven, now considering the required parallelism, which traduces in hardware replication. In the eighth chapter some additional algorithms are presented, one to estimate the SNR of the signal after equalization, and the other to perform a phase error correction. Finally, in the ninth chapter, some conclusions and suggestions of possible future work are presented.

1.6. Author publications

As an outcome of some of the work performed, two publications in conference proceedings were made.

1. P.S. Costa, M.V. Drummond, R.S. Ribeiro, C. Oliveira, N. Ribeiro, R. Nogueira and P.P. Monteiro, “Experimental evaluation of a polarization tracking algorithm for single-polarization M-(D)PSK signals using coherent detection”, in International Conference on Transparent Optical Networks, Warwick, 2-5 July 2012.
2. P.S. Costa, M.V. Drummond, R.S. Ribeiro, C. Oliveira, N. Ribeiro, R. Nogueira and P.P. Monteiro, “Evaluation of a polarization tracking algorithms for single-polarization M-(D)PSK signals using coherent detection”, in Symposium on Enabling Optical Networks and Sensors 2012.

2. Optical coherent systems

Depending on the technique used to receive the modulated information, receivers may have two different architectures: direct detection (DD) or coherent detection. In DD the receiver does not track the phase of the incoming light beam, merely performing the reception based on the intensity of the received light. The received signal is directly converted to baseband through the low pass characteristics of the photo-detector. Phase difference between symbols can also be detected using optical differential detection [20]. Here, there is no need for a local oscillator (LO), resulting in cheaper receivers where the sensitivity is independent of the carrier phase and the state of polarization (SOP) of the received optical signal [4]. This technique, however, performs a nonlinear optic to electric conversion, in which both phase and polarization information are lost, reducing the capabilities of posterior DSP.

The other option is to use coherent detection, where the received electrical signal is proportional to the electrical field of the received light, down-converted in frequency. Incoming light is “mixed” with a local oscillator and the resulting signal will be centered at an intermediate frequency (IF), corresponding to the difference between frequencies of both oscillators. If both frequencies are the same, the resulting signal will be in baseband and the system is called homodyne. If IF is not zero, the receiver is called heterodyne. This last architecture requires, at least, twice the bandwidth in the receiver but does not require narrow linewidth and phase locked lasers [12], resulting in cheaper systems. Furthermore, in a WDM network, a heterodyne receiver allows the down-conversion of several WDM channels into different intermediate frequencies, which can then be received and processed in the digital domain simultaneously.

2.1. Modulation formats

To communicate through an optical fiber, the information must change the properties of the generated laser light beam. The basic modulation formats affect the amplitude (ASK), phase (PSK), frequency (FSK) or even the polarization of the light. These formats do not use the available bandwidth in an efficient way. Another option is to use advanced modulation formats, where multiple dimensions of light are modulated. For example, it is possible to combine amplitude and phase modulation in two different data streams and send them through the same fiber using polarization division multiplexing (PDM).

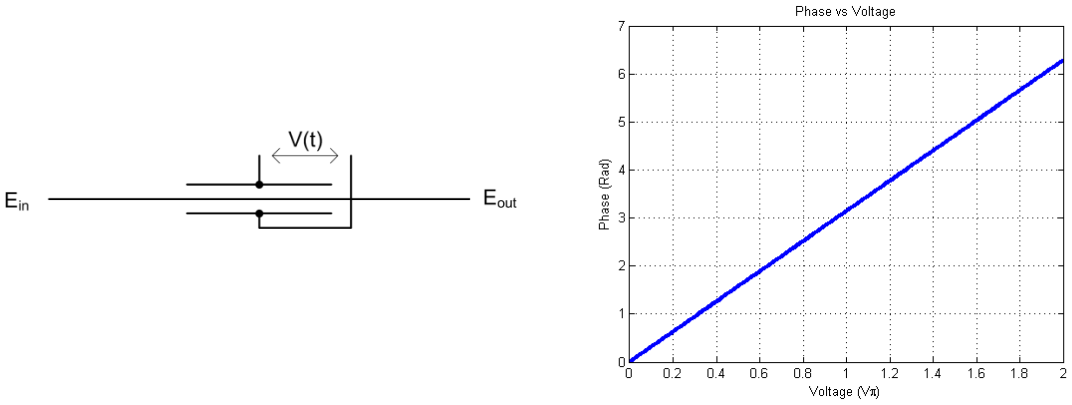
The electrical signal containing the information must be used to modulate the laser. This can be achieved through direct modulation or using an external modulator. In direct modulation the electrical signal containing the information is used to operate the laser, generating IM (on-off keying (OOK)). This produces cheap transmitters but the variation in the bias current produces considerable pernicious phase modulation, labeled as chirp [21], which make the transmission of high data rates impracticable. The alternative is to use external modulation, which allows from intensity modulation (for example the electro-absorption modulators (EAM)) to intensity and phase modulation (the electro-optic modulators (EOM)).

The simplest EOM is the phase modulator (PM), created by inserting an electro-optical substrate in the optical waveguide. By changing the applied external voltage it is possible to change the refractive index of the waveguide, inducing different speeds in the light, which results in different phases of the output optical signal (relative to the input). The phase at the output of a PM is proportional to the applied voltage and the required voltage to produce a phase rotation of π radians is called $V\pi$. The output signal is given by

$$E_{out}(\mathrm{t}) = E_{in}(\mathrm{t})e^{j\frac{\pi \mathrm{V(t)}}{V_\pi}}, \tag{2.1}$$

where V represents the applied voltage.

The PM and its phase response are displayed in Figure 2.1.



a) b)

Figure 2.1 a) Phase modulator and b) phase response with applied voltage (normalized to $V\pi$)

By applying different voltages it is possible to control the phase of the output signal. Considering that certain groups of bits lead to specific voltages, resulting in specific phases at the output, it is possible to represent the information as symbols

$$S(k) = e^{j\varphi(k)}. \tag{2.2}$$

The simplest modulation format produced with a PM is the BPSK, where the output phase shifts from 0 to π, corresponding to 0 and 1. Increasing the number of different symbols more bits can be assigned to each symbol ($N_{bits} = log_2(N_{symb})$) resulting in a lower symbol rate for the same data rate, which increases the spectral efficiency [22].

In the case of QPSK modulation, it is possible to represent the symbols by

$$S(k) = \sqrt{P_0}\mathrm{e}^{j\varphi(k)}, \quad k = 0,1,2,3 \tag{2.3}$$

where P_0 is the power of the laser and $\varphi(k)$ is obtained considering Table 2.1.

Information	00	01	10	11
Symbol k	0	1	2	3
Phase $\varphi(k)$	0	$\frac{\pi}{2}$	$-\frac{\pi}{2}$	π

Table 2.1 Phase for each symbol of a QPSK constellation considering Gray coding

Some PSK constellations are shown in Figure 2.2.

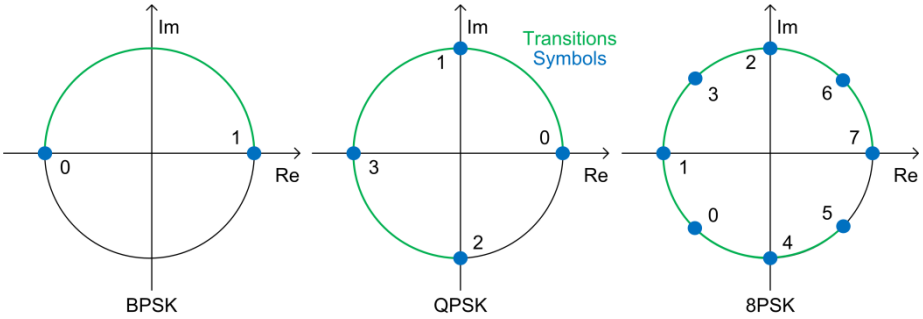
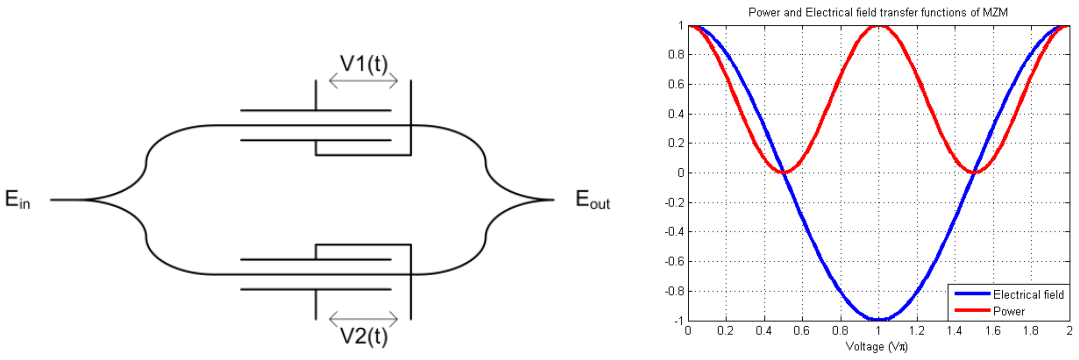


Figure 2.2 Different PSK constellations

The main advantage when using PM is that the output power is always constant, even when symbol transitions occur (if NRZ is used). However, the fast inter-symbol transitions result, through the modulator, in a fast phase rotation in the optical signal, producing an undesired frequency modulation, called transient chirp [20]. As a result, the optical signal will have a larger bandwidth, which may impose limits in the symbol rate (due to dispersion) and channel spacing [1].

Using two PM in parallel it is possible to obtain a Mach-Zehnder modulator (MZM). The incoming light is split in two paths, and each one is modulated using a different voltage. The resulting signals are then combined. By controlling the applied voltages it can produce constructive or destructive interference at the output. If the applied voltages are symmetric, $V_2 = -V_1$, it is possible to control the power of the laser and perform intensity modulation. This mode of operation, represented in Figure 2.3, is called push-pull and can be expressed by

$$E_{out}(t) = \frac{E_{in}(t)}{2} cos\left(\frac{V(t)}{V_\pi}\pi\right). \quad \textbf{(2.4)}$$



a) b)

Figure 2.3 a) MZM modulator and b) MZM transfer function in push-pull configuration

By using two Mach-Zehnder modulators it is possible to obtain an IQ (from In-phase and Quadrature) modulator, illustrated in Figure 2.4. The optical beam is split in two paths, where the two signals are modulated using MZMs. In one of the paths an additional phase shift of 90º is introduced, originating the quadrature component.

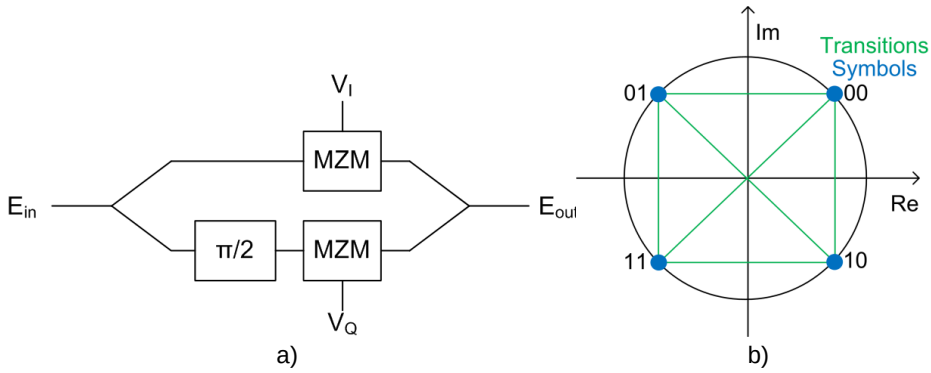


Figure 2.4 a) IQ modulator and b) 4QAM constellation

Again, the resulting signal can be expressed by symbols, this time in the form of

$$S(k) = I(k) + jQ(k). \tag{2.5}$$

Controlling the applied voltages it is possible to produce any desired symbol, not necessarily containing unitary amplitude as it was obtained using the PM.

In the previous modulation formats, the information was encoded in the transmitted symbols. However, it is possible to transmit information in the transitions between symbols. In this case the modulation format is additionally called differential. The absolute phase state φ_k is determined by the desired phase difference φ_{diff} and the previous phase, φ_{k-1}, through $\varphi_k = \varphi_{k-1} + \varphi_{diff}$, as shown in Figure 2.5.

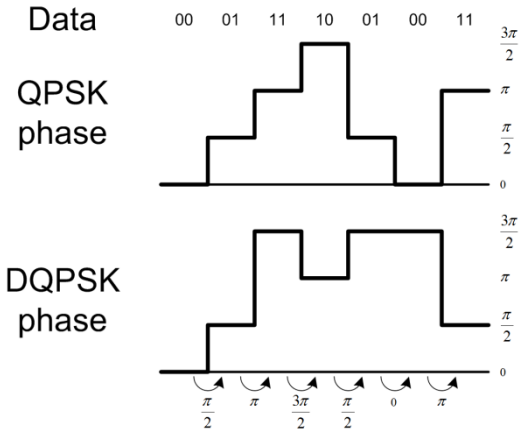


Figure 2.5 Comparison between QPSK and DQPSK

Differential modulation has the disadvantage of presenting a SNR penalty at the receiver (in the case of DQPSK modulation, it has a SNR penalty of $2.3dB$ when compared with QPSK [23]). However, it allows the simplification of the receiver since the received signal is allowed to have an additional phase rotation arising, for example, from a mismatch in the phase between the oscillators in the transmitter and the receiver. Since only differential phase is detected, this phase mismatch is removed while, in non-differential modulation, this phase error must be tracked and corrected.

2.2. DQPSK transmitter employing a PM

The block diagram for a DQPSK transmitter is presented in Figure 2.6.

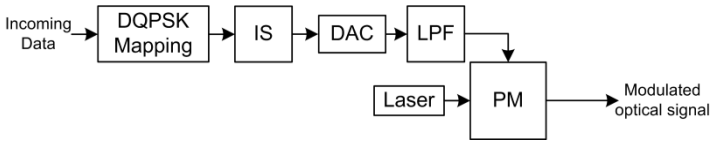


Figure 2.6 Block diagram of a DQPSK transmitter using a PM

At the DQPSK mapping block the incoming data is analyzed using blocks of 2 bits (since each symbol carries 2 bits) and the correct symbols are generated, according to the incoming data and to the last sent symbol. The determined symbols then go through an impulse shaper (IS), where the signal is given the desired shape, for transmission and bandwidth purposes. After digital to analog conversion the signal is applied an analog low pass filter and the resulting voltages can then be applied to the PM.

2.3. Fiber propagation

The optical fiber is not an ideal waveguide, and the optical signal suffers distortions while traveling through it. There are different sources for this degradation, and they can be grouped in two major categories, the linear and nonlinear effects.

2.3.1. Linear effects

Fiber Attenuation

By propagating in the optical fiber, the optical signal loses some power, majorly due to material absorption and Rayleigh scattering [1]. The power can be expressed by

$$P(L) = P_{in}e^{-\alpha L} \tag{2.6}$$

where α represents the absorption coefficient, usually expressed in dB/km, and L is the traveled distance. In standard single mode fiber (SSMF), $\alpha \approx 0.2dB/km$. The attenuation, along with the power of the laser at the transmitter and the sensibility of the receiver will contribute to limit the maximum length of the network and to the maximum splitting. Attenuation can be compensated by optical amplification.

Chromatic Dispersion

When travelling in an optical fiber, different frequencies have different speeds, which results in different times of arrival, spreading the optical pulses. This factor limits the maximum data rate as inter-symbol interference starts to occur.

There are two sources of dispersion. Material dispersion takes place due to the dependence of the refractive index of the fiber with the frequency, $\eta(w)$, which results in frequency-dependent speeds. The second source for dispersion is waveguide dispersion, and is related with the power distribution inside the fiber. Small wavelengths are confined

to the core of the fiber, presenting $\eta_{eff} \approx \eta_1$ while larger wavelengths have a considerable power traveling in the cladding, presenting $\eta_{eff} \approx \eta_2$ [24].

These two effects combined are called chromatic dispersion (CD). It is represented in units of $ps/(nm.km)$. The accumulated dispersion indicates the maximum difference between the times of arrival for the frequency components of a given channel, being more severe to high bandwidth signals and large distances. In SSMF the total dispersion is $D = 17ps/(nm.km)$ at $\lambda = 1.556\mu m$.

It is possible to obtain a frequency domain transfer function that describes the chromatic dispersion [14], representing the CD as a simple filter. Through DSP the CD can be compensated in the received signal by applying the inverse filter.

Let us consider a QPSK channel operating at $622MSym/s$ (bandwidth of $1.244GHz$) over a distance of $100km$. For this case, the total dispersion will be $D = 17ps$. The symbol duration is $1.6ns$, which means that, in this case, the total dispersion represents 1% of symbol duration, a value so small that, at the receiver, the distortion induced by CD is negligible. For this system, no CD compensation is required. This will be the case of NGOA, introduced in chapter 3.

Polarization Mode Dispersion

Before referring to PMD there is interest in clarifying some topics about the polarization of light. Even in a SSMF it is possible to identify two orthogonal modes of propagation. The electrical field can be divided into two arbitrary orthogonal components, called E_{ox} and E_{oy}. These two components don't have necessarily the same power or phase. They can be represented by

$$E_{ox}(z,t) = \hat{e}_x\sqrt{P_{ox}}\,cos(wt - \beta z)$$
$$E_{oy}(z,t) = \hat{e}_y\sqrt{P_{oy}}cos(wt - \beta z + \delta) \quad \textbf{(2.7)}$$

where $|E| = \sqrt{P_{ox} + P_{oy}}$ is the amplitude of the total electric field.

Depending on δ, E_{ox} and E_{oy}, it is possible to identify the different SOP, illustrated in Figure 2.7:

- $\delta = 0 + 2m\pi$, m=1, 2, …, originate linear polarization
- $\delta = \frac{\pi}{2} + 2m\pi$, m=1, 2, …, and $E_{ox} = E_{oy}$ originate circular polarization
- Other values originate elliptical polarization

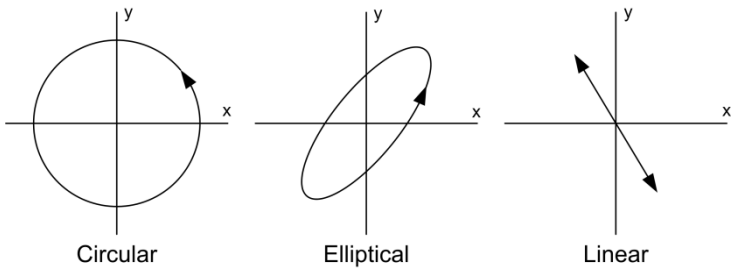


Figure 2.7 Different possible SOP

The optical signal is launched in a fiber with a random orientation relative to the fiber's axes of polarization, giving rise to both polarizations. If the fiber was a perfect cylinder, these two modes would coexist independently. In real fibers, however, the symmetry is not perfect due to applied external forces, impurities introduced in the fabrication or temperature oscillations [25]. This asymmetry results in birefringence: the difference in the refractive indexes of both orthogonal polarizations, which induces different propagation speeds. This produces a fast and a slow mode, leading the light to go through all the polarization states [26].

In small sections of fiber, birefringence can be considered uniform and the difference between propagation constants for the fast and slow modes is given by [26]

$$\beta_s - \beta_f = \frac{w}{c}\left(\eta_s - \eta_f\right) = \frac{w}{c}\Delta_\eta \tag{2.8}$$

where η_s e η_f represent the effective refractive indexes for the slow and fast modes, respectively, w is the angular frequency and c the speed of light. This velocity difference will introduce a differential group delay (DGD), $\Delta\tau$, that is proportional to the traveled distance since the birefringence is considered constant. The DGD has dependence with the wavelength. In channels operating with small bandwidths, the DGD can be considered constant across each WDM channel, but different from channel to channel [27]. In this case, the PMD is called first order PMD. However, when operating with high data rates, the effects of the DGD dependence with the frequency cannot be neglected, as it originates effects similar to chromatic dispersion [28], which means that, for these cases, higher order PMD must be considered. In this work, only first order PMD is considered, since each channel has small bandwidth.

Presently there are fiber spans with tens or hundreds of kilometers. These can be studied as a concatenation of a large number of small fiber sections, each one with its own birefringence axes, represented through a Jones Matrix

$$T_n = \begin{bmatrix} u_{1,n} & u_{2,n} \\ -u_{2,n}^* & u_{1,n}^* \end{bmatrix} = B_n R_n \tag{2.9}$$

where B_n represents the birefringence matrix for the n_{th} segment

$$B_n = \begin{bmatrix} \mathrm{e}^{\mathrm{j}\frac{\theta_n}{2}} & 0 \\ 0 & \mathrm{e}^{-\mathrm{j}\frac{\theta_n}{2}} \end{bmatrix}, \tag{2.10}$$

θ_n is the phase difference resulting of the DGD, and $R_n(\partial)$ is the rotation matrix, representing a random coupling angle between adjacent fiber segments

$$R_n = \begin{bmatrix} \cos\left(\partial_n\right) & \sin\left(\partial_n\right) \\ -\sin\left(\partial_n\right) & \cos\left(\partial_n\right) \end{bmatrix}. \tag{2.11}$$

Note that in (2.9) $\left|u_{1,n}\right|^2 + \left|u_{2,n}\right|^2 = 1$, as expected from the rotation matrix R_n.

This means that there is a random rotation in the polarizations axes whenever the light goes from one section to the next. One polarization in one section may originate two components, each one in a different polarization, in the next section. The total fiber, considering N sections, can be represented by [29]

$$T = \begin{bmatrix} u_1 & u_2 \\ -u_2^* & u_1^* \end{bmatrix} = \prod_{n=1}^{N} T_n. \quad \textbf{(2.12)}$$

For long fiber lengths the cumulative DGD follows a Maxwellian distribution and depends on the square root of the traveled distance. In the presence of high symbol rate the DGD extends over a significant part of the symbol duration, which must be compensated or will result in a high inter-symbol interference. In [9] an average DGD close $0.1ps/\sqrt{km}$ is indicated, and DGD lower than this value has been measured in fibers deployed after 2000. Considering the nominal value of $0.1ps/\sqrt{km}$, in a fiber with $100km$ the expected DGD will be around $1ps$. Considering again a system operating at $622MSym/s$ this represents a very small fraction of the symbol duration meaning that, as far as to the receiver concerns, the DGD will result in a simple phase rotation between both polarizations that needs to be compensated.

When polarization multiplexing (PDM) is used, PMD will scramble both signals, which will have to be demultiplexed at the receiver. However, when using single polarization, the total first order PMD effects can be simplified. The total effects, in (2.12), when applied to a singly polarized signal originate

$$\begin{bmatrix} E_{ox} \\ E_{oy} \end{bmatrix} = \begin{bmatrix} u_1 & u_2 \\ -u_2^* & u_1^* \end{bmatrix} \begin{bmatrix} E_{os} \\ 0 \end{bmatrix} = \begin{bmatrix} u_1 E_{os} \\ -u_2^* E_{os} \end{bmatrix} \quad \textbf{(2.13)}$$

where E_{ox} and E_{oy} represent, respectively, the optical x and y polarizations after the PMD effects. Knowing that $|u_1|^2 + |u_2|^2 = 1$ and considering the phase of E_{oy} as a reference it is possible to rewrite (2.13) as

$$\begin{bmatrix} E_{ox} \\ E_{oy} \end{bmatrix} = \begin{bmatrix} \sqrt{\rho} e^{j\delta} \\ \sqrt{1-\rho} \end{bmatrix} E_{os} \quad \textbf{(2.14)}$$

where $\delta = \angle(u_1) - \angle(-u_2^*)$.

These steps show that, at the receiver, only the power split ratio ρ and the phase difference between polarizations δ need to be estimated to revert the effects of PMD.

Due to temperature changes and mechanical vibrations, the PMD and DGD change over time. Normally these parameters originate slow variations that go from hours to minutes. In tests performed in deployed fibers, PMD fluctuations were measured to time intervals of milliseconds [30]. However, stress tests show that PMD can suffer variations in microseconds [31].

In addition to PMD introduced by the fiber, the optical components, although having a constant birefringence, introduce themselves a random rotation that, combined with the other elements, will contribute to the total PMD effects at the output signal [25].

2.3.2. Nonlinear effects

In a WDM network all the clients share the medium simultaneously. This means that the fiber carries a vast number of wavelengths, confined to a small region that is the fiber core, resulting in a high power density. If high enough, this power can change the refraction index of the fiber, introducing phase distortions in the transmitted signal [27]. The correction of nonlinear effects is not the purpose of this dissertation, but a small reference to the major sources of nonlinear interference is given in the next subchapters.

Self phase modulation

Self phase modulation (SPM) occurs when the power of a propagating signal is enough to change the refractive index of the fiber and induce phase distortions in the signal itself, leading to pulse broadening [32]. When the power envelope of the emitted signal is constant, for example if PSK NRZ modulation is used and the effects of CD are negligible, the resulting SPM will induce a constant phase rotation. However, if amplitude modulation is used, in the worst case OOK modulation or RZ impulse shape, the induced phase change will depend on the signal.

Cross phase modulation

Cross phase modulation (XPM) has the same principles of SPM applied to a WDM network, where the power of a propagating channel may introduce phase distortions in the other WDM channels [1]. Again, if PSK is used in every WDM channels and the effects of CD are negligible, the total power envelope is constant and will result in a constant phase shift. However, if one of the channels uses amplitude modulation, all the other channels will be affected.

Four wave mixing

Four wave mixing (FWM) is a major source of inter-channel crosstalk in a WDM network when the channel spacing is small. The interaction of different wavelengths in a nonlinear medium originates new harmonic wavelengths. If a constant spacing between channels is used the resulting wavelengths may coincide with pre-existing WDM channels, causing interference. The effects of FWM are intensified when the propagating signals are phase matched, which happens if low-dispersion fibers are used [33]. For SSMF the presence of CD mitigates the phase matching and reduces the effects of FWM [2].

2.4. Receiver

When the optical signal has information in its phase and the SOP of the incoming light is random, a heterodyne receiver comprising phase and polarization diversities should be used. Before presenting such receiver, its constituent components are introduced in the following subchapters.

Coupler 3dB

Obtained through the fusion of two fibers, a coupler 3dB equally divides the power of the input fibers by the outputs. A coupler 3dB can be expressed through

$$\begin{bmatrix} E_{out_1} \\ E_{out_2} \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & j \\ j & 1 \end{bmatrix} \begin{bmatrix} E_{in_1} \\ E_{in_2} \end{bmatrix}. \quad \textbf{(2.15)}$$

Balanced photodetector

To perform the conversion from the optical to the electrical domain photodetectors are used, converting the power of the electric field into current. If the incoming optical signal is previously combined with the local oscillator, the result will be centered at the IF. The use of two paired inversely polarized photodetectors, called a balanced photodetector (BD), doubles the received power and allows cancelation of local oscillator intensity noise, the main advantage from the use of BD [34]. The output current of a balanced photodetector is given by [20]

$$I_{out}(t) = r.E_1(t)E_1^*(t) - r.E_2(t)E_2^*(t) \quad \textbf{(2.16)}$$

where E_1 and E_2 are the input electrical fields of the optical signals received in the upper and lower arms, respectively, and r is the responsivity of the photodiodes.

Polarization beam splitter

When arriving at the receiver, the optical signal has a random SOP. To perform the beat between the local oscillator and the received signal, both signals must have the same polarization. This means that the two polarizations present in the signal should be separated and received independently.

A polarization beam splitter (PBS) is a passive component that is able to generate two linearly polarized orthogonal components out of a signal with a random SOP. The separation occurs when the light beam goes through the interface of two materials. One of the polarizations will be reflected while the other propagates through the interface. The polarizations are separated using a random direction, meaning that the obtained polarizations are not necessarily the two polarizations present in the fiber, but a mix between them.

Heterodyne receiver with phase diversity

A simple heterodyne receiver with phase diversity, similar to the one presented in [9], is present in Figure 2.8. Since heterodyne reception is used, the received signal is down-converted to an intermediate frequency and, in the digital domain, the phase diversity is obtained.

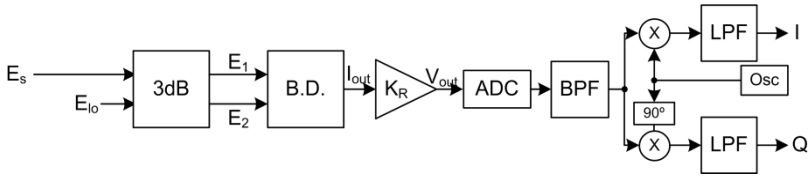


Figure 2.8 Heterodyne receiver comprising phase diversity

Let's consider that $E_s = \sqrt{P_s}e^{j(w_1 t+\theta s+\phi)}$ is the electrical field of the incoming modulated optical signal, where θs represents the phase modulation, ϕ expresses a phase error and the electrical field of the local oscillator is given by $E_{lo} = \sqrt{P_{lo}}e^{jw_2 t}$. At the output of the BD, and considering that the electric components present a small bandwidth (enough to allow the channels on their intermediate frequencies, $\Delta = w_1 - w_2$, but eliminating the remaining high frequency beatings), the generated current will be

$$I_{out} = 2r\sqrt{P_s P_{lo}}sin(\Delta \mathrm{t} + \theta s + \varphi). \quad \textbf{(2.17)}$$

By controlling the power of the local oscillator it is possible to amplify the received signal, one of the advantages to the use of coherent detection. If N WDM channels are considered the received current will be

$$I_{out} = 2r\sqrt{P_{lo}}\sum_{k=1}^{N}\sqrt{P_{s,k}}\, sin(\Delta_k \mathrm{t} + \theta s_k + \varphi_k), \quad \textbf{(2.18)}$$

where each signal is given a specific IF.

This current is then converted into voltage using a transimpedance amplifier and the signal is sampled. The sampling rate must be, at least, twice the highest IF plus half of the channel bandwidth considered to preserve the Nyquist limit for every channel. A band pass digital filter, centered in the desired IF, is then applied to filter noise and remove other WDM channels, reducing inter-channel interference. Using a local electrical oscillator at IF and low pass filters, to eliminate the resulting high frequency components, it is possible to down-convert the in-phase and quadrature signals to baseband, $I \propto cos(\theta s + \phi)$ and $Q \propto sin(\theta s + \phi)$ or, combining both signals, $e^{j(\theta s+\varphi)}$, the signal containing the phase information. Note that the phase error introduced is present in the received signal. In this case, since DQPSK modulation is desired, this phase does not represent a problem if small variations are allowed, since it is removed in the demodulator. However, if QAM or PSK modulation are used, this phase error must be tracked and corrected [13]. After the down conversion, DSP can be applied to perform signal equalization.

By changing the frequency of the local electrical oscillator and the central frequency of the band pass filter, a different WDM channel can be received. A system can be set up to receive several channels simply by using different parallel electrical receivers after the ADC stage.

Heterodyne receiver with phase and polarization diversities

To characterize the SOP of the incoming light, the receiver must be doubled and the polarizations separated. The local oscillator must be polarized at 45 degrees so that the

power is equally distributed and both polarizations of the incoming signal are received. The complete receiver is presented in Figure 2.9.

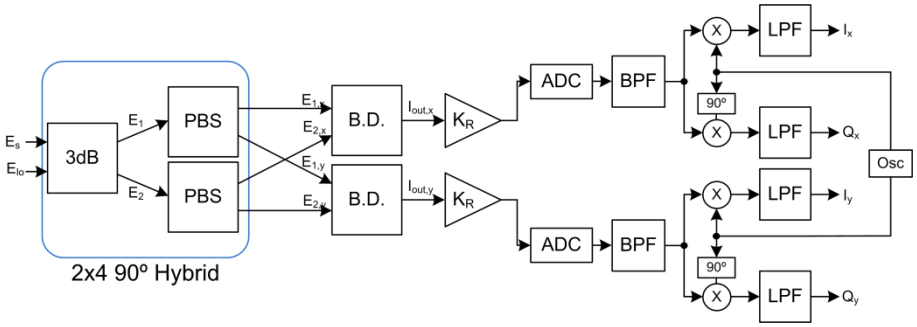


Figure 2.9 Heterodyne receiver comprising phase and polarization diversities

After combining the local oscillator and the incoming light in the 3dB coupler, the polarizations of both outputs are separated through PBSs, and the resulting signals are received using BDs. Each polarization is then received independently, and combined through DSP to extract the transmitted information.

This architecture is able to receive PSK and QAM formats, since all the information present in the electrical field of the optical signal is received. This linear reception enables full equalization of the signal through DSP:

3. Next Generation Optical Access

The increasing need for bandwidth, combined with an increasing number of broadband services users, has been pressing service providers towards more efficient access networks. Ultra dense wavelength division multiplex (UDWDM), combined with advanced modulation formats has proven to be the best option to build a network that is future proof. One of the possible solutions is the next generation optical access (NGOA), proposed by Nokia Siemens Network. Based on an UDWDM, passive optical network (PON), and coherent detection, NGOA enables an unprecedented split ratio (up to ~1000), a very long unamplified reach (~$100km$), and symmetric unshared $1Gb/s$ rates per user.

3.1. Passive optical networks

The optical access network connects the clients to the service provider. It is comprised by the optical line terminal (OLT), located at the service provider, by the optical network units (ONU), located in the clients and by the optical distribution network (ODN), the network itself. It is distinct from the core network, used to transport high rate traffic between service providers.

A passive optical network (PON) is an optical network with point to multipoint architecture, where the ODN does not need power supply. This factor imposes a limitation to the maximum distance but allows the construction of cheaper networks, since no power line is needed. PON are generally built using a tree topology, allowing a fast adaptation to the increasing number of clients [7].

3.2. Architecture of NGOA

Being NGOA based in WDM architecture, all the clients have permanent access to the medium. The channel grid is imposed by the OLT, where the downstream wavelengths are generated. Each ONU then tunes its local oscillator using one of the wavelengths (with an offset of Δ, which corresponds approximately to $1GHz$), different from the others to avoid collisions [35]. The channels are grouped in optical transmission groups (OTG) and, between OTGs, a band guard of $50GHz$ avoids inter-group interference. The channels at each OTG are disposed as presented in Figure 3.1.

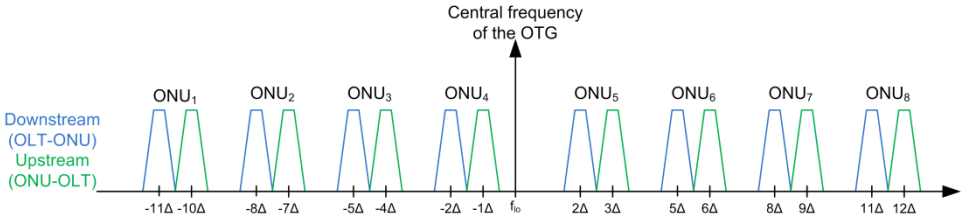


Figure 3.1 OTG spectra [36]

In order to reduce the costs of implementation, NGOA is designed to run in the currently deployed networks meaning that, instead of using WDM splitters such as arrayed waveguide grating (AWG), conventional passive splitters are employed. This

factor introduces high attenuation, limiting the maximum distance in the network and the number of users, but leads to a more flexible architecture towards the increase in the number of clients. A generic WDM network implemented using a PON with a tree topology is presented in Figure 3.2.

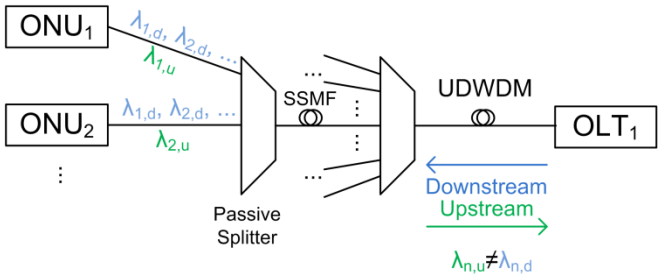


Figure 3.2 WDM network with tree topology

Several levels of splitting must be used so that all the clients can access the network. The position and number of splitters will have considerable impact in the network since they will reduce the optical power, contributing to a reduction of the impact of the non-linear effects and limiting the maximum distance in the network.

When connecting to the network, each ONU must first lock its LO in the desired channel through the downstream information sent by the OLT, producing a stable oscillator. The oscillator is modulated using DQPSK to generate the upstream channel. The same laser is simultaneously used to perform the downstream demodulation, using heterodyne reception with IF=Δ, the frequency difference between the upstream and downstream channels relative to that ONU. By means of DSP the receiver is able to retrieve the information even when the local oscillator has a shift of 50MHz to the correct value, which relaxes the requirements of the phase locked loop (PLL) used for the LO [35]. The block diagram of an ONU is present in Figure 3.3.

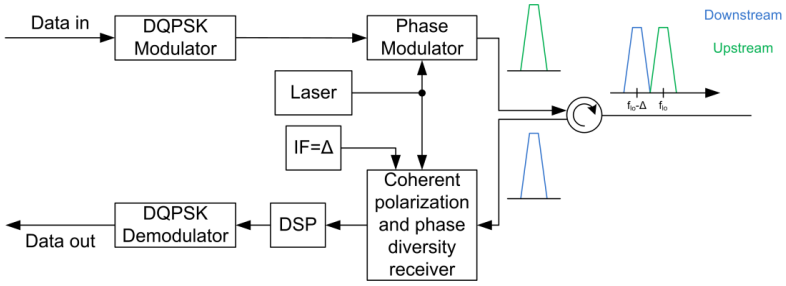


Figure 3.3 Block diagram of a ONU [37]

The OLT, Figure 3.4, located at the service provider, must be able to receive and send the information from and to a large number of ONUs. In order to reduce the required hardware, all the channels present in each OTG are received simultaneously using a single LO (corresponding to the central frequency of the OTG) through heterodyne reception. After mixing the incoming light with the LO, an OTG presents the spectra presented in Figure 3.5.

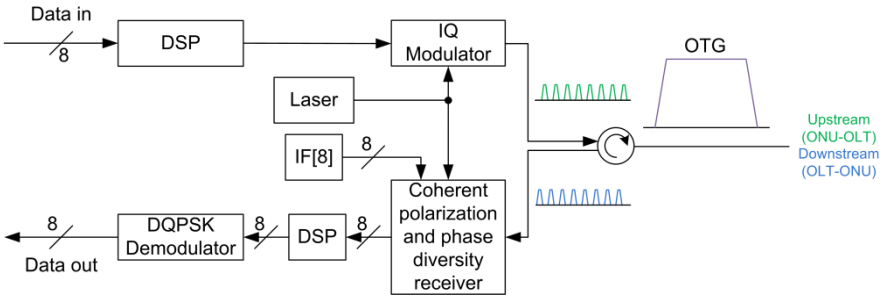


Figure 3.4 Block diagram of the OLT

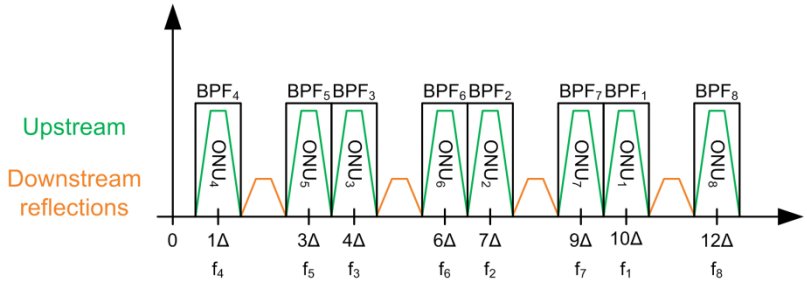


Figure 3.5 Received spectrum at the OLT [38]

After the down conversion at the OLT, DSP is applied to compensate the propagation impairments for each channel before retrieving the information from each client. The downstream information is modulated using the same laser. The different frequency channels are firstly generated in the digital domain, though digital multi-carrier modulation. These will then originate the different optical channels through an IQ modulator.

3.3. Problem to solve and specifications

The incoming light at the OLT has a random SOP. In order to receive the total information present in the optical domain, a polarization and phase diversity receiver must be used. Due to fiber impairments, the SOP of each channel m can be described as two components, E_{ox} and E_{oy}, that relate as

$$\begin{bmatrix} E_{ox,m} \\ E_{oy,m} \end{bmatrix} = \begin{bmatrix} \sqrt{\rho_m} e^{j\delta_m} \\ \sqrt{1-\rho_m} \end{bmatrix} E_{os,m}. \quad (3.1)$$

These parameters are independent for each channel, since each channel goes through different paths, even if only the span between the ONU and the first splitter. An example for the received signals is given in Figure 3.6.

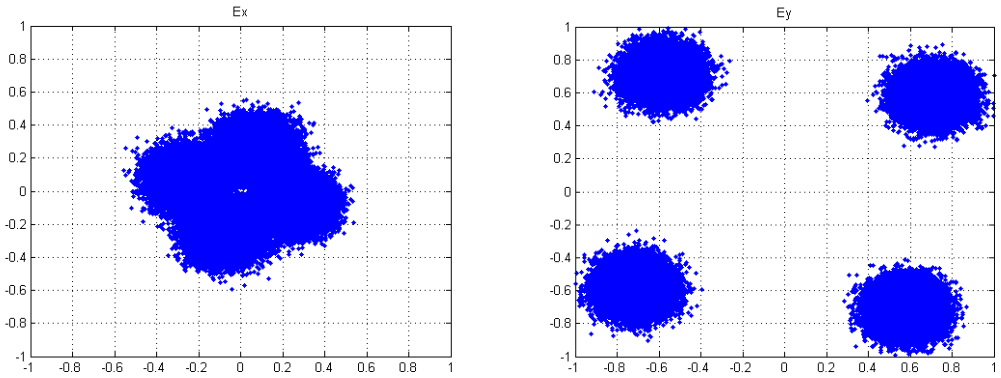


Figure 3.6 Received constellations when $\rho = 0.4$ and $\delta = \pi/3$ (fixed parameters to simplify the plot)

Since these parameters are unknown and change over time, the received polarizations cannot be added directly after detection, since they could combine destructively. These parameters must be corrected independently for each channel, which makes the compensation in the optical domain impracticable due to the required amount of hardware. The feasible solution is then the use of DSP to estimate and compensate these parameters. Hardware replication will be required for the different channels.

A simple method to extract the information would be selecting the polarization with the highest power. However, if the power is equally distributed by both polarizations, only half of the received power would be used, and noise would have high impact in the output signal. Also, since the phase difference between polarizations changes in time, phase jumps could occur when switching between polarizations.

This problem can be corrected using an adaptive algorithm to track both ρ and δ, in order to align the phase of both polarizations and perform a maximal ratio combiner, trying to reduce the impact of noise. No training sequences are used, which means that blind polarization tracking must be performed. The block diagram of the receiver is shown in Figure 3.7.

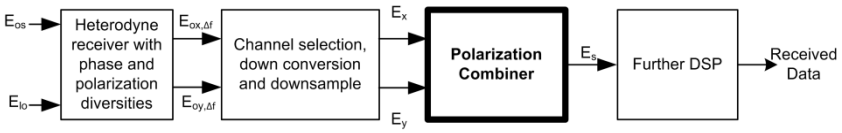


Figure 3.7 Block diagram of the receiver

Some specifications were given to perform this task. Each client will have available symmetric unshared $1\ Gb/s$ connections. Introducing the forward error correction (FEC) code and other additional information to perform the overhead, a bit rate of $1.24416\ Gb/s$ for each client is obtained. Using DQPSK modulation the symbol rate is $622.08\ \mathrm{MSymb/s}$. A pre-FEC limit of $BER = 10^{-3}$ is imposed in order to efficiently retrieve the data from the received signal.

At the OLT, after performing the heterodyne reception, filtering and down-converting the received signal, the resulting baseband signal is sampled using $2\ samp/Sym$, at a rate of $1.24416\ GSamp/s$, and represented in signed two's complement fixed point [39] using 8 bits, in the format Q.7 (one signal bit, zero integer bits and 7 fractional bits). These samples will be used to perform, amongst other operations, polarization tracking and alignment. The used hardware is designed to work at $155.52\ MHz$, meaning that 8

samples (respecting 4 symbols) must be processed simultaneously (the degree of parallelization (DOP) will be 8) for each channel.

Due to hardware budget, it is required that five channels are processed using the same Virtex6 FPGA. This means that five different phase differences and SOP must be tracked and $8 * 5 = 40$ samples must be processed and compensated in every clock cycle.

Field tests performed indicate that, in the worst case scenario, PMD effects can suffer variations close to $50kHz$ [31]. In practical cases however, a variation of some kHz should be considered, situation that is closer to the measures performed in [30].

4. Development environments

4.1. Behavioral simulation

The tests for the different algorithms always start with a behavioral simulation. In the first simulations performed to the different algorithms an ideal system has been considered, where all the incoming samples are analyzed, operating at one sample per symbol and taking the ideal sampling point. This way, a symbol-wise test is performed, generating one symbol per simulation cycle. These conditions are useful to determine if the algorithm presents the desired behavior or not.

4.1.1. MATLAB and SIMULINK

MATLAB, “Matrix Laboratory”, is a high level language used for numerical computation. The basic data element in MATLAB is the matrix and it allows fast programming and computation when manipulating array-based data. It can be used to develop algorithms, models, simulate behaviors and analyze data. MATLAB has a vast number of libraries that allows it to connect to different external peripherals or programs. Due to its simplicity and performance when operating with large arrays, and simplicity in the RS232 connection used for communication in the hardware section later on, MATLAB was selected to perform the data generation and analysis.

SIMULINK is an add-on product to MATLAB. It has a simple and easy-to-use graphical interface that allows building models through block diagrams. Simulations can be performed and the results analyzed in run-time using numeric or graphic outputs. Due to its flexibility and performance SIMULINK was selected to design and simulate the desired algorithms.

4.1.2. Simulation scheme

Before developing and testing the polarization tracker algorithms, we must have blocks to generate the information, simulate the desired channel interference and perform the reception. These blocks are explained in the next subchapters.

Transmitter

The first block used in simulations is the DQPSK transmitter. Considering a symbol-wise simulation, the symbols are directly generated using a random integer generator from 0 to 3, corresponding to 00, 01, 10 and 11, respectively. During the entire simulation the information is carried as symbols and only in the post-simulation analysis bit-wised information would be required to perform a BER analysis, but the same result can be obtained by symbol analysis. The DQPSK transmitter used is shown in Figure 4.1.

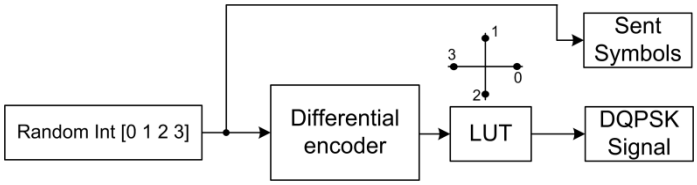


Figure 4.1 Block diagram for the DQPSK transmitter

In the Differential encoder block, the differential information is produced from the incoming and the previously sent symbols. After this step, the DQPSK constellation is created using a look up table (LUT).

Channel simulator

The objective of the circuit to be designed is the capability to estimate the power split ratio (ρ) and the phase difference (δ) between both polarizations and, with this information, restore the received signal. To assess the capability of the circuit for following the variation of the parameters ρ and δ, these are simulated by two sinus waves, with frequencies slightly different so that the phase between both factors varies over time, thus analyzing all possible situations. In the worst case scenario, the maximum frequency of the sinusoids is specified to be close to $50kHz$, the maximum frequency of variation measured to the SOP of an optical signal. Operating at $155.52MHz$ this translates into

$$\frac{2\pi*50*10^3}{155.52*10^6} = \frac{\pi}{1555.2} = 6.4 * \pi * 10^{-4} rad/sample. \quad (4.1)$$

Both channels are also corrupted with noise. AWGN is used to simulate the noise introduced in the received baseband components of each polarization. Considering a given signal power, several values for the noise power are used to investigate the circuit's response with different input signal-to-noise ratios (SNR). The additive noise is independent for each polarization and also independent for the real and imaginary components. Both polarizations will be affected with the same noise power. The circuit used to simulate the channel is represented in Figure 4.2.

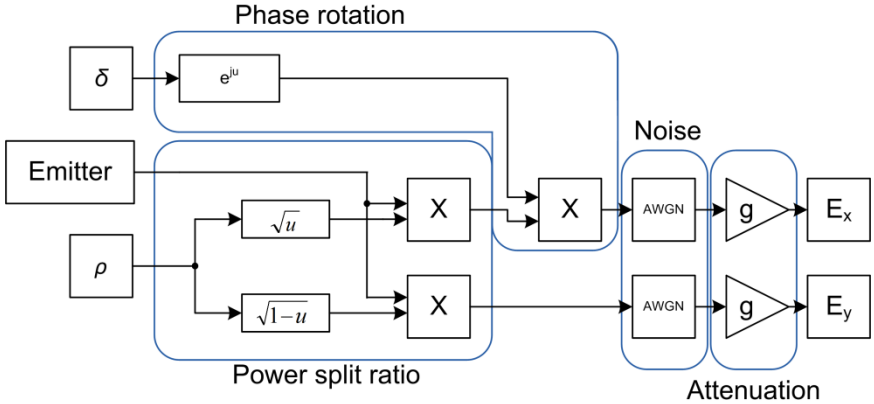


Figure 4.2 Block diagram for the channel simulator

The phase difference is only introduced in one of the polarizations. However, since the phase relation is estimated and compensated between both polarizations, the circuit would have the same behavior if both polarizations were affected.

In a particular case there will be interest in analyzing and compensating the amplitude of the signal. To test this case a gain factor g is introduced for both polarizations ($g < 1$ introduces attenuation), and its value is simulated using a sinusoid with very slow oscillations, to determine if the circuit is able to work properly when the received power varies in time.

Receiver

There are several ways of assessing the performance of the circuit. Different figures of merit are available, such as error vector magnitude (EVM), SNR penalty or bit error ratio (BER) of the output signal. The EVM is estimated by comparing the received signal with the ideal signal [40] and, in some of the designed circuits, a phase drift will be allowed for some particular conditions, meaning that this measure would lead to incorrect results. A BER analysis was selected to perform the comparison between the different algorithms, as it allows determining if instantaneous phase jumps occur and allows estimating of the SNR penalty introduced, without the penalty due to the phase offset introduced, since it is removed in the DQPSK demodulation.

To perform a symbol and / or binary analysis of the circuit, the receiver's mode of operation is represented in Figure 4.3.

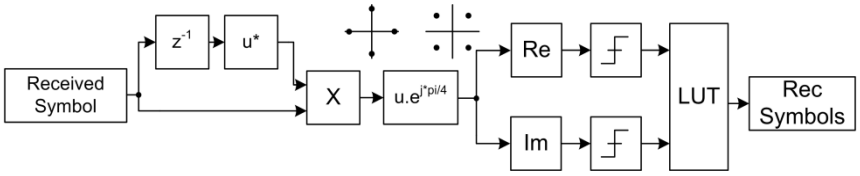


Figure 4.3 Block diagram for the DQPSK receiver

To perform the demodulation of the received signal, the arriving symbol is multiplied by the complex conjugate of the previously received symbol, and the resulting angle is the phase difference between the two adjacent symbols, the differential phase containing the information. This signal has a phase of 0, $\frac{\pi}{2}$, π or $\frac{3\pi}{2}$, so a $\frac{\pi}{4}$ rotation is performed so that the symbols are positioned in the middle of each quadrant and, simply by evaluating the signal's real and imaginary components, it is possible to obtain the carried information. Using a zero-crossing detector for each component (real and imaginary) and, by accessing to a LUT, it is possible to determine the received symbol (0, 1, 2 or 3). These symbols are saved to the workspace so that they can be compared with the transmitted information.

Binary and symbol analysis

Since symbol wised simulations are performed, it is only possible to compare both emitted and received symbols. However each symbol carries specific binary information. With this in mind it is possible to estimate the number of bit errors simply by comparing the stored emitted and received symbols. Performing a Monte Carlo (MC) analysis a BER value is then obtained. The first 1000 symbols (2000 bits) of each simulation are not used because they correspond to the startup situation, where the circuit is adapting to the new input signals, and it is desired to analyze the circuit in a continuous mode of operation.

Performance limit

To test the simulation blocks, a simple BER test can be performed. From [23] it is possible to obtain the approximate error probability to a DQPSK system when considering uncorrelated noise, as is being considered in this case,

$$P_e \approx \frac{1}{2} erfc\left(\sqrt{\frac{1.1716}{4}\frac{E_S}{N_0}}\right). \quad \textbf{(4.2)}$$

These values can be compared with the simulation results to evaluate if the designed systems are working properly by considering that all the received power is in one of the polarizations, and use that as the input for the receiver. The results obtained are shown in Figure 4.4.

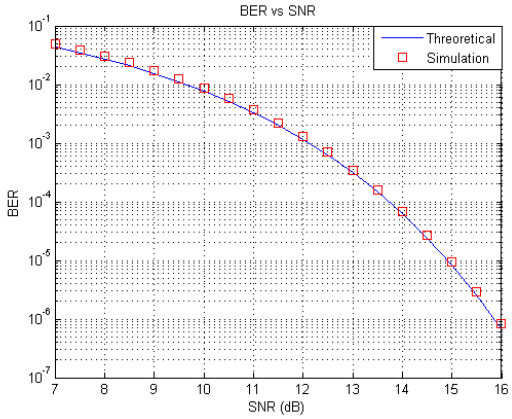


Figure 4.4 BER vs SNR curve for DQPSK modulation

The obtained results are consisted with the expected values, proving that the circuit has the desired behavior.

As seen before, the noise is considered to have the same power in both polarizations but the receiver should be able to, through the compensations, “see” only half of the total noise, similar to the previous situation where only one polarization, with the total signal power but only half of the total noise power, is received. This means that the curves determined before will be improved by a factor of 3dB. It is possible to obtain the limit BER vs SNR curve in Figure 4.5, which will be achieved if the circuit used for compensation has an ideal performance.

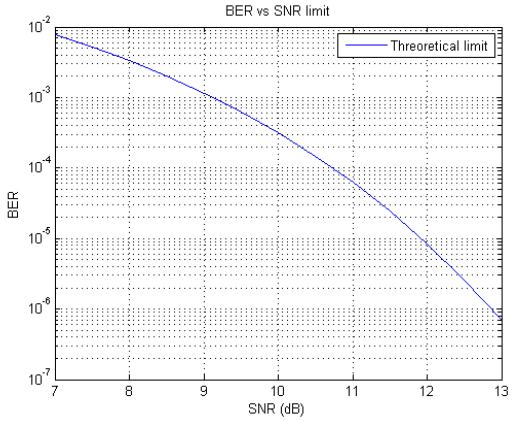


Figure 4.5 Limit BER vs SNR curve for the dual polarization DQPSK system

Even though a short range of SNR values is represented, the interesting case of study is the situation where the BER is close to 10^{-3}, the FEC limit. This curve will be compared with the obtained results from the different circuits to test their performance.

The $BER \leq 10^{-3}$ limit is presented by DQPSK modulation for a total input SNR between $9dB$ and $10dB$. With this in mind, the tests performed to study the performance of the circuit while estimating both ρ and δ will be performed considering a SNR of $10dB$. A standard worst case scenario can be set up to test the circuit's performance:

- $0 < \rho < 1$ and frequency near $50kHz$;
- $-\pi < \delta < \pi$ and frequency near $50kHz$;
- $SNR = 10dB$.

There is a circuit (CMA) that is unable to cope with such fast oscillations. To test this circuit, the same conditions are considered, but now the frequencies are set to be close to $800Hz$.

Influence of delay in parameters estimations

Before designing the circuit, it would be interesting to study the influence that the delay introduced by the circuit under test in the estimated parameters has in the output signal. Since the circuit is to be implemented in hardware, and hardware circuits introduce delays in the processed signals, as will be seen later, some signal degradation may arise from the compensation using delayed parameters. The circuit used to study this problem is presented in Figure 4.6.

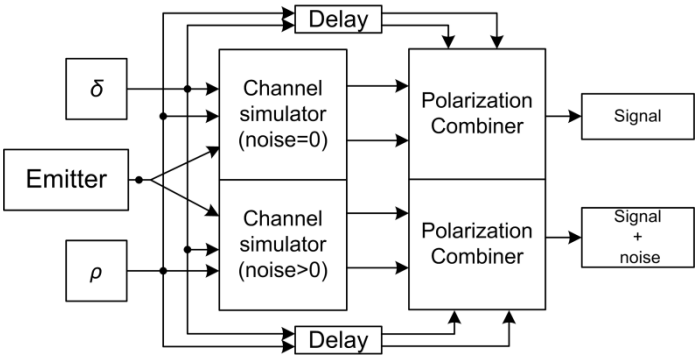


Figure 4.6 Setup used to study the influence of delay in the estimated parameters

In the polarization combiner, considering an ideal behavior, the steps are the inverse of those applied in the channel simulator, using the ideal parameters with a delay. By having two parallel systems, one with and one without noise, it is possible to obtain, at the output, the compensated ideal signal and the compensated signal with noise. By subtracting both results it is possible to obtain the noise component and thus, having also the output noise free signal, estimate the SNR. This imperfect compensation will lead to a decrease in the signal power, but the noise will keep the same total power, leading to a decrease in the SNR, or a SNR penalty.

Since the parameters were simulated by sinusoids, it is possible to express the delay in radians. If we do so, the SNR penalty obtained will be independent of the frequency of variation considered for ρ and δ. The penalty is also independent of the considered SNR, since the penalty is due to a decrease in the signal power.

The results obtained are presented in Figure 4.7.

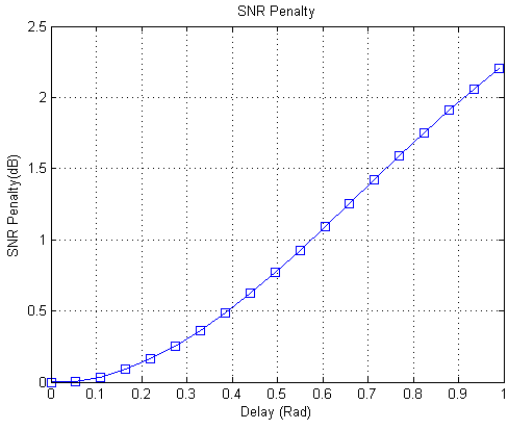


Figure 4.7 SNR penalty for an introduced delay in the estimated parameters

It is possible to conclude that, while for small delays the SNR penalty introduced is small, as the delay increases a considerable penalty starts to appear. For this reason, the worst case scenario will be the one where both ρ and δ have the maximum frequency of oscillation, situation where, due to the delays introduced by the circuit, the delay in radians is maximum.

4.2. Real case simulation

In a real case scenario 2 samples per symbol are used. This means that one of the samples may correspond to a symbol transition. Also, while the tests performed to the algorithms suppose AWGN in the received components, this does not represent accurately the reality. To test the last circuit under real conditions, where fiber effects and inter-channel interferences are taken into account, an optical simulator was used to generate the data that was then processed by the circuit. In this case, the ability of the circuit to estimate the desired parameters is tested in the presence of interference. A BER analysis could lead to erroneous conclusions since now several random interferences are present.

4.2.1. OSIP

OSIP [41] is an optical simulation platform developed at Universidade de Aveiro and Telecommunications Institute. It has a drag-and-drop graphical interface that simplifies the description of optical systems. OSIP is open source and based in MATLAB, meaning that any person with knowledge in MATLAB can inspect the content of each block and even create and add new blocks.

4.2.2. Optical setup

In OSIP a system was designed, trying to meet some of the characteristics presented by the NGOA network. The conceptual network is presented in Figure 4.8. No BER analyses are performed using OSIP, as different sampling points will be considered and the sequences of generated symbols are small to obtain good results. Instead, the

performance of the circuit while estimating the desired parameters will be studied, now in the presence of interferences close to the ones that occur in a real optical network.

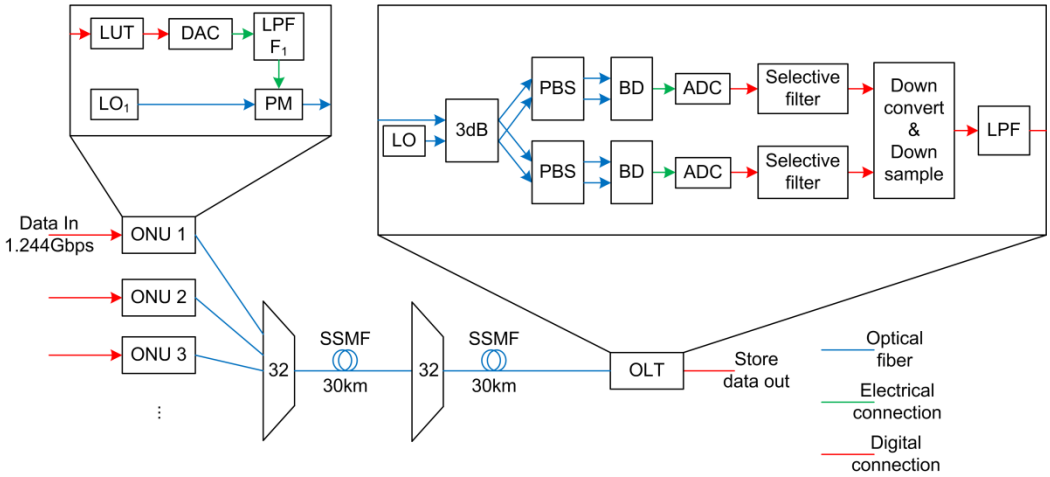


Figure 4.8 Optical network designed in OSIP

At the emitter a DQPSK constellation is generated through a PM. A Bessel low pass filter with $1GHz$ bandwidth is applied to the electrical signal. Only five ONUs were considered, to study the inter-channel interference resulting from the small frequency spacing. The frequencies for the different oscillators are:

- $F_{OLT} = 193.5THz$,
- $F_{ONU_1} = 193.499THz$,
- $F_{ONU_2} = 193.503THz$,
- $F_{ONU_3} = 193.496THz$,
- $F_{ONU_4} = 193.506THz$,
- $F_{ONU_5} = 193.493THz$.

In the optical network two splitting stages are used, with $30km$ SSMF connections, where both linear and non-linear effects enter into account. Both lasers are considered to be ideal, so that no random phase rotations occur. Even if phase mismatches occur, the tested circuit would not see them, since it only corrects a phase difference between two received polarizations, δ, and any laser mismatch would introduce a phase error simultaneously in both polarizations.

At the receiver, the ADCs automatically adjust the dynamic range to the input amplitudes so that all the information is received. An ideal rectangular filter is considered with a bandwidth of $600MHz$ for the baseband digital signals in an attempt to reduce inter-channel interference. The outputs are then normalized (considering the maximum power peak) and stored for posterior analysis.

The spectrum received at the OLT is represented in Figure 4.9, where the frequencies are relative to the LO of the OLT. Note that the different channels are positioned in the same frequencies of the upstream channels shown in Figure 3.1.

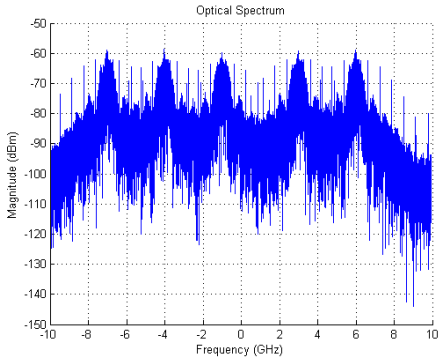
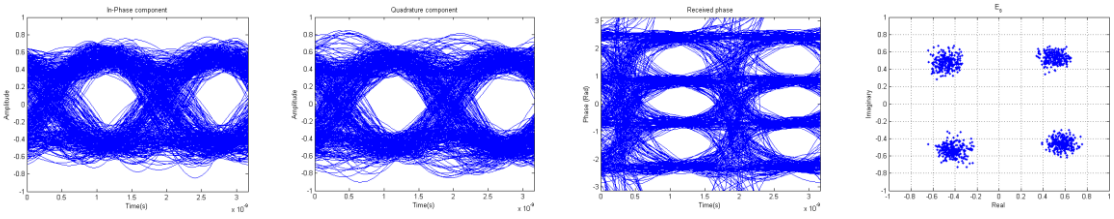


Figure 4.9 Received spectrum at the OLT

Different channels could be received and analyzed. After the down conversion in the heterodyne receiver, some channels are spaced only by $1GHz$. For example, the channel centered at $3GHz$, in Figure 4.9, will be received next the channel centered at $-4GHz$ in the same figure (after the heterodyne down-conversion, this channel will be centered at $4GHz$), as seen in Figure 3.5. Due to higher inter-channel interference, one of these channels (the one centered at $3GHz$) was chosen and used in the tests.

In a first test no PMD was considered, simply trying to analyze the circuit performance. The eye diagram for the received in-phase and quadrature components and for the encoded phase can be seen in Figure 4.10, where data previous to down sampling is considered. Also, considering an ideal sampling point, the received constellation is represented.

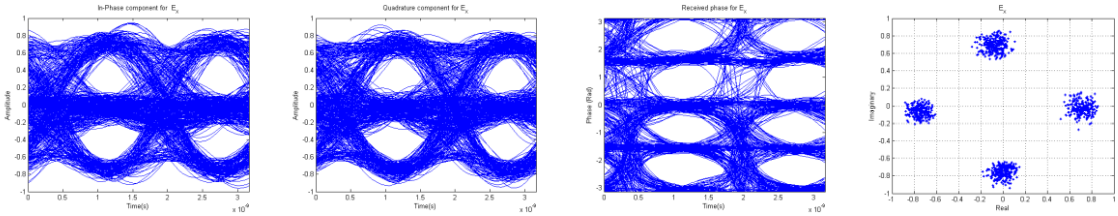


a) b) c) d)

Figure 4.10 Eye diagrams for the received signals (a) and b)), c) encoded phase information and d) resulting constellation

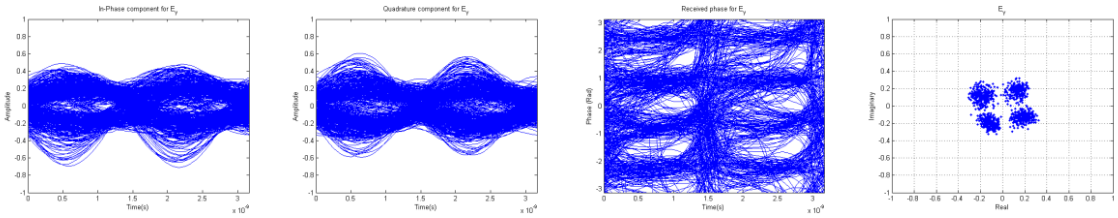
From the eye diagrams it is possible to see that the received signals present high jitter, resulting from the intense filter, and high noise due to inter-channel interference, resulting from the small channel spacing. Despite these factors, the phase information can be easily extracted if the optimum sampling point is selected. In a real implementation of the circuit the received signal should present a quality similar to that considered in these simulations, meaning that, if the circuit performs well under these conditions, it will work in the real case.

The next test repeats the same procedures, now considering PMD. The PMD effects were simulated independently for each channel, as it would happen in a real implementation, since different channels go through different fibers, even if it is just the span used to connect the ONU to the first passive splitter. The results obtained for the two polarizations are presented in Figure 4.11 and Figure 4.12. Again, no down sampling was considered when tracing the eye diagrams and the ideal sampling point was considered to plot the constellations.



a) b) c) d)

Figure 4.11 Eye diagrams for the received signals for the E_x polarization (a) and b)), c) encoded phase information and d) resulting constellation



a) b) c) d)

Figure 4.12 Eye diagrams for the received signals for the E_y polarization (a) and b)), c) encoded phase information and d) resulting constellation

It is possible to see that the PMD effects are characterized as a power distribution between both polarizations, ρ, and a relative phase rotation, δ. OSIP introduces a constant phase rotation, meaning that the circuit won't be tested with varying parameters. Nevertheless, this data represents a good scenario in terms of the interferences and distortions the optical signal would face in a real optical system. The results obtained for the in-phase and quadrature components of both polarizations, shown in Figure 4.11 and Figure 4.12, will be down sampled to two samples per symbol and used to test the last algorithm, in conditions close to those present in a real case implementation.

4.3. Hardware setup

The developed algorithms have the last objective of being "translated" into VHDL and implemented in a FPGA, where they will process the received signals in real time. For that reason, before explaining the hardware scheme used to test the algorithms a brief introduction to the used hardware and programming tools is performed.

4.3.1. FPGAs and Evaluation kits

A FPGA (field programmable gate array) can be seen as an array containing thousands of logic blocks that can be programmed and interconnected as desired by the developer. Each basic block, conceptually represented in Figure 4.13, is constituted by LUTs, multiplexers, flip-flops (FF) and additional logic, and can perform simple tasks. In the Virtex FPGA family, these blocks are called Slices. Slices are then grouped to create configurable logic blocks (CLBs) that can be used to perform many different tasks [42].

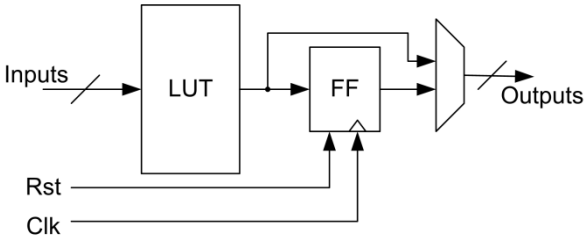


Figure 4.13 Block diagram of a generic Slice

In order to be flexible the connections between the logic blocks must be fully reprogrammable, leading to small frequencies of operation. However, due to the large number of available resources, FPGAs can be used to perform parallel processing, by which it is possible to obtain higher throughputs than those obtained using a pipelined processor [43]. The high data rates used in optical communications used to be a limitation to the use of FPGAs. However, due to new advances in CMOS technology higher clock frequencies are now possible and a renewed interest has been put in the use of FPGAs to perform DSP in optical systems [44].

Nowadays FPGAs are, in fact, more complex than a simple array of logic cells, including dual port RAMs, clock management units, flexible I/O interfaces, DSP cells to perform fast multiplications, communication ports and much more. In this work a Virtex6 XC6VLX240T-1FFG1156 FPGA is used, where each slice is constituted by four six-input LUTs, multiplexers, four FF/Latches and four FF. The most relevant characteristics of the Virtex6 are present in Table 4.1.

Logic cells	Slices	Distributed Ram (Kb)	DSP48E1 Slices	Block RAM blocks			Transceiver GTX	Ethernet MACs
				18Kb	36Kb	Max(Kb)		
241.152	37.680	3.650	768	32	16	14.976	24	4

Table 4.1 Virtex6 XC6VLX240T FPGA summary [45]

The FPGA itself is nothing but a small chip. To communicate with the outside and simplify the debugging, FPGAs are often integrated into evaluation kits, such as the ML605. These evaluation kits facilitate the communication and configuration of the FPGA and often have important features. In the case of ML605, some of the key features are [46] (see also Figure 4.14):

- FPGA Virtex6 XC6VLX240T-1FFG1156;
- Configuration through USB2.0 or Flash memory;
- Power supplied through transformer;
- DDR3 (512MB) and Flash memories (34MB+16MB+8Mb);
- 200MHz differential oscillator and 16MHz socket clock;
- SMA sockets for external clock or user GPIO;
- LCD display, LEDs, Pushbuttons and Switches;
- Communication through RS232, Ethernet, USB2.0, GTX and PCI Express.

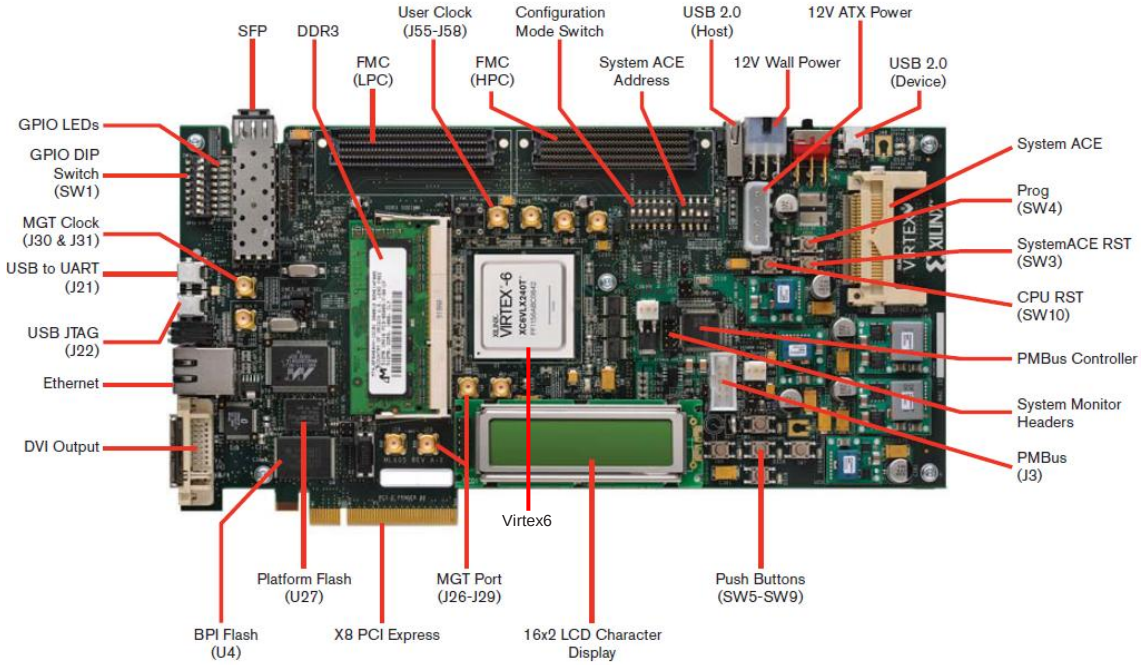


Figure 4.14 ML605 [47].

4.3.2. **Hardware description language**

Configuring the connections and logic blocks “by hand” in a FPGA would require high knowledge, skill and time even for the most experienced designer. Instead, the circuit can be designed through schematic editors or using hardware description languages (HDL), where is possible to specify the behavior of the circuit, by describing the outputs as a function of the inputs for each circuit. While in normal programming languages the instructions are processed sequentially in a processor, HLDs support parallelism, meaning that a high number of independent processes can run simultaneously.

The HDL description of the circuit allows not only the hardware implementation but also behavioral simulations, which simplifies the designer’s tasks as the hardware implementation requires much more time than simulation. However, simulation by itself does not ensure that the same circuit will behave correctly in reality, so hardware tests must be performed.

The two main HDLs are VHDL and Verilog. VHDL was selected to perform the hardware implementations.

4.3.3. **Integrated synthesis environment**

After writing the VHDL code it must be corrected and converted into the desired digital circuits. Xilinx ISE 13.3 was used to perform this task. When developing a VHDL project there are some steps to follow before obtaining the circuit operating in a FPGA.

After writing the VHDL code, the first step performed by the ISE is the synthesis of that code, generating the hardware and interconnections that implement the modeled behavior. Syntax errors present in the code are detected at this point and must be

corrected before moving to the next steps. Also, at this point the designer can perform a behavioral simulation, testing the outputs for each possible binary input.

In addition to the VHDL code, the designer must elaborate a constraint file where the timing requirements and input/output ports positions are specified. This can be used to connect the control signals to the switches and the outputs to the LEDs of the evaluation kit, or to ensure that certain signal has a maximum delay, important to clocks propagation. This constraint file is used in the next step, the implementation, where the generated netlist is mapped into the FPGA, allocating resources at particular locations and routing the interconnections between them. The output of this step is the FPGA configuration file and a report containing the used resources, timing and power estimations. At this time a more realistic simulation can be performed, testing the timing of the internal signals. When performing the implementation step, the ISE performs intensive calculation so that the chosen location for the hardware ensures the timing constraints, the reduction of hardware resources and the minimum power consumption.

The last step is, using the configuration file, program the FPGA and test the performance of the circuit while operating in real hardware.

Additionally to the debugging and compiler capabilities, the Xilinx ISE also provides language templates, to help in the creation of simple hardware structures, and intellectual property cores, IPCores [48], to perform specific functions. While the templates provide only the basic frames, which must be completed and customized by the designer, the IPCores generate specific hardware where the designer simply needs to indicate the number of inputs and outputs and other desired performance factors. When possible these cores should be used, since they result in a more efficient implementation.

4.3.4. **Hardware test platform**

After simulating the behavior of the circuits, some of them were implemented in a FPGA to test their performance under real hardware limitations. The transmitter, channel and receiver were implemented in MATLAB and are conceptually similar to the block diagrams presented in the simulation subchapter. The biggest difference is that the circuit under test is now implemented in hardware. To receive and transmit the data from and to the circuit, a RS232 interface was created in MATLAB to communicate with the development kit. In the FPGA it is also necessary to use a RS232 interface to receive and transmit the data from and to the PC. The used RS232 module is an open core, and can be seen as a black box. The complete test circuit implemented is shown in Figure 4.15 a).

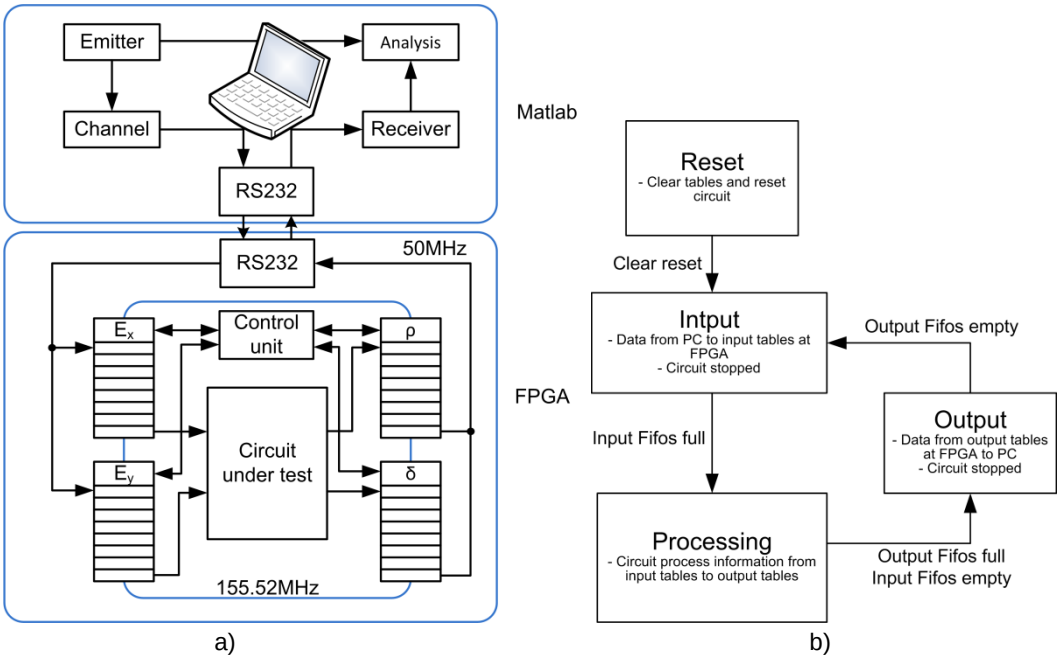


Figure 4.15 a) Hardware test environment and b) finite state machine used

A Moore Finite State Machine [49], presented in Figure 4.15 b), was implemented to synchronize the FIFOs, the RS232 interface and the circuit. The communications from and to the PC are preformed using blocks of 257 Bytes for each component, the size of the FIFOs used by the RS232 interface, obtained using the “Fifo generator 8.3” [50]. By stopping the circuit during the communications it is ensured that, as far as the circuit is concerned, the information is arriving and leaving sequentially, being insensitive to the context switch happening in the system.

After an initial state of reset, the circuit goes into the Input state, where it is ready to receive information. Here, the PC transmits the data that fills the input tables while the circuit is stopped. When the input Fifos are full, the circuit goes to the Processing state, where the communications are disabled and the circuit processes the samples stored in the input memories, filling the output Fifos. When the output Fifos are full (input Fifos empty) the circuit is stopped and the processed information is transmitted to the PC. After this step the circuit returns to the Input mode, expecting more information from the PC. Error correction codes could be used to implement a more complex and reliable communication protocol, instead of relying only in the information from the Fifos. However RS232 proved to work properly without these additional tools.

The circuit is intended to work at $155.52MHz$ inside the FPGA. However, due to synchronism and hardware complexity, the RS232 interface was designed to operate at $50MHz$. Using the “Clock Wizard 3.2” [51], the two different clocks, operating at $50MHz$ and $155.52MHz$, were generated from the $200MHz$ differential clock available in the ML605. The Fifos were designed to operate using different clock frequencies for the input and output, and accommodate the clock difference.

Although the information in MATLAB is processed with high precision, the communications with the FPGA are made in the format Q.7, as specified. Inside the circuit a different representation is used to ensure that little information is lost during intermediate calculations. This will be explained in more detail in chapter 6.

5. Algorithms and Simulations

5.1. Algorithms used for dual polarization

When equalizing effects such as PMD, where the induced distortions are time-varying, adaptive algorithms are used. Such algorithms are commonly based in adaptive FIR filters, where the taps are updated using an error signal, obtained from the output signal through a feedback loop. In Figure 5.1a) is represented the block diagram for a generic adaptive FIR filter and, in Figure 5.1b), a MIMO equalizer with a butterfly structure is presented.

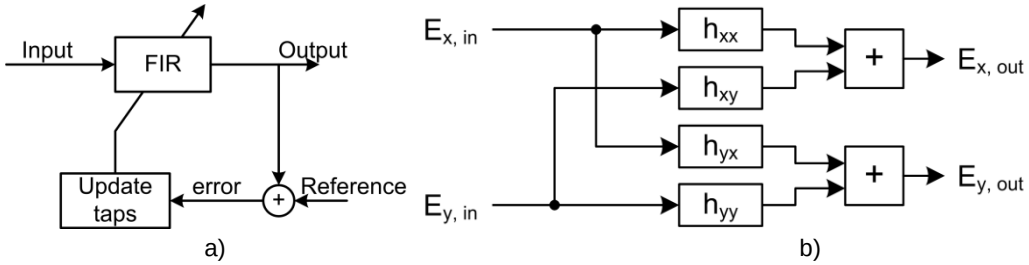


Figure 5.1 a) Generic block diagram for an adaptive FIR filter and b) butterfly-structured MIMO equalizer

In the butterfly MIMO filter of Figure 5.1b), the outputs are

$$E_{x,out} = E_{x,in} * h_{xx} + E_{y,in} * h_{xy}$$
$$E_{y,out} = E_{x,in} * h_{yx} + E_{y,in} * h_{yy} \quad \textbf{(5.1)}$$

where * designates convolution between the two signals.

There are two similar algorithms, the constant modulus algorithm (CMA) and the decision-driven least means square (DD-LMS), that are implemented using the presented butterfly MIMO architecture to perform compensation for PMD effects. The CMA has the best performance and requires fewer resources to be implemented. However, it has a slightly worse response for higher frequencies [14, 52]. The DD-LMS, on the other hand, is able to equalize signals with (slightly) faster variations in the SOP, but requires a decision circuit to estimate the error. The CMA algorithm was selected and tested, since it is the algorithm with the best performance and implementation.

5.1.1. Constant modulus algorithm

In the CMA, the filters are updated using the following expressions.

$$h_{xx} = h_{xx} + \mu\, \kappa_x\, E_{x,in}\, E^*_{x,out}$$
$$h_{xy} = h_{xy} + \mu\, \kappa_x\, E_{y,in}\, E^*_{x,out}$$
$$h_{yx} = h_{yx} + \mu\, \kappa_y\, E_{x,in}\, E^*_{y,out}$$
$$h_{yy} = h_{yy} + \mu\, \kappa_y\, E_{y,in}\, E^*_{y,out} \quad \textbf{(5.2)}$$

where μ is a small positive gain and

$$\kappa_x = 1 - \left|E_{x,out}\right|^2$$
$$\kappa_y = 1 - \left|E_{y,out}\right|^2 \quad \textbf{(5.3)}$$

are used to normalize the power of the outputs, where '1' (the constant modulus) is the reference signal used to estimate the error.

In this dissertation, no PDM is used, which means that a single polarization will be obtained at the output, and the other output will be zero. This allows a simplification in the circuit to obtain Figure 5.2, where only two filters and one output signal are necessary. Additionally, only κ_x (the power of the output) needs to be estimated.

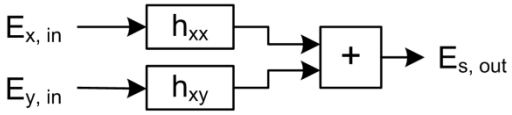


Figure 5.2 CMA used when no PDM is used

Also, in the considered optical system, the optical channel does not present memory, since dispersive effects can be neglected when compared with the large symbol duration, as seen in chapter 2. This means that the filters h_{xx} and h_{xy} can be reduced to a single tap, where the equalization is simplified to an amplification factor and a phase rotation.

Let us remember the received signal after the PMD-induced distortions, (2.14).In this case, since the output power will be normalized, it is expected that the circuit will converge to a situation where a maximal ratio combiner is performed, since the output power will be normalized and

$$\sqrt{\rho}\left(\sqrt{\rho}E_x\right) + \sqrt{1-\rho}\left(\sqrt{1-\rho}E_y\right) = E_s \quad \textbf{(5.4)}$$

which is normalized if E_s presents a constant unitary power envelope, such as the DQPSK (considering, for this case, that the input signal does not suffer from attenuation). It is then possible to see that the amplitude of the taps will inadvertently have the estimated amplitude of the input signals, $\sqrt{\rho}$ and $\sqrt{1-\rho}$. More on maximal ratio combiners will be given in the next chapter, where the principles and advantages are explained.

Also, to allow the convergence of the output signal, both E_x and E_y must have the same phase after equalization. This means that the phase difference δ can be obtained by estimating the phase difference between both taps h_{xx} and h_{xy} after convergence.

5.1.2. **Circuit and simulation**

After the previous simplifications to the CMA algorithm, the resulting circuit, comprising both the filters and the feedbacks used to update the taps, is represented in Figure 5.3.

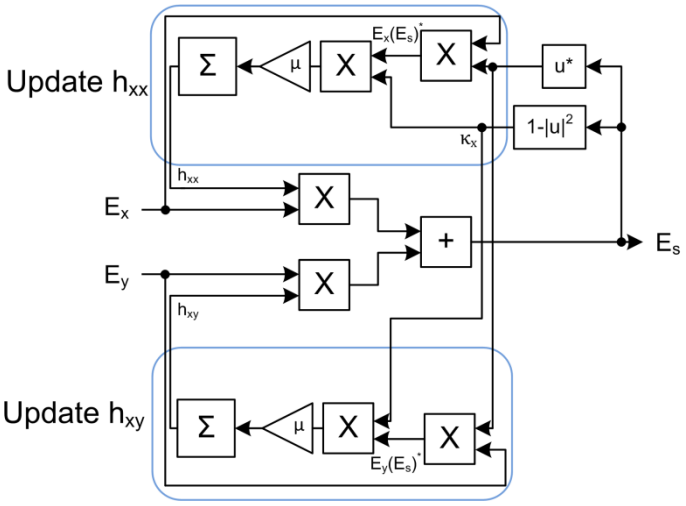


Figure 5.3 Block diagram for the implementation of the CMA

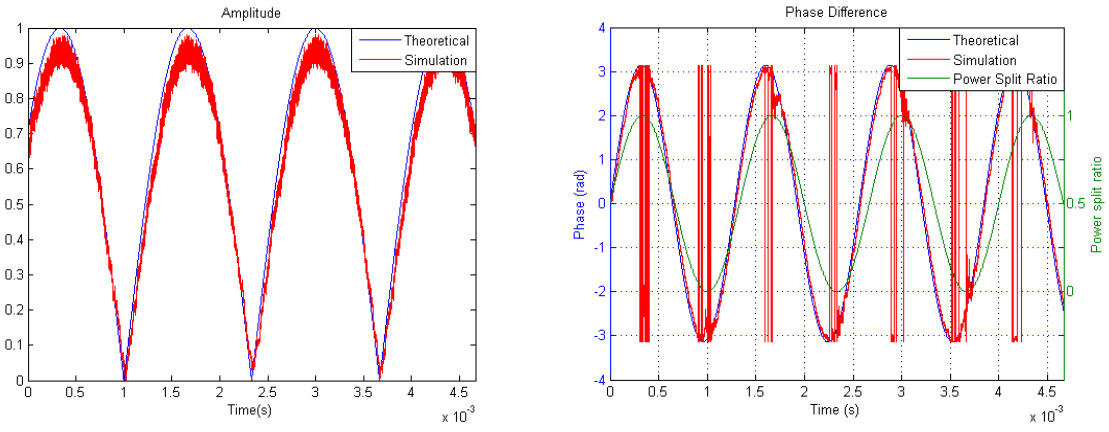
After some trials, $\mu = 10^{-2}$ was selected, presenting a good behavior in the presence of noise. This value is in the normal range used for the iteration factor [52]. However, as a result of this low value, the circuit will have a slow response, being able only to track low frequencies of variation in the SOP of the received signal.

In this case, a frequency of oscillation for ρ and δ close to $800Hz$ was selected to test the circuit, close to the maximum frequency the circuit is able to track without introducing considerable penalty. The estimations of these parameters can be obtained by analyzing h_{xx} and h_{xy}, since, after convergence,

$$e^{j\delta} = h_{xx}^* h_{xy}$$
$$\sqrt{\rho} = |h_{xx}| \text{ (or } \sqrt{1-\rho} = |h_{xy}|), \tag{5.5}$$

as explained before.

The obtained results are presented in Figure 5.4. In this case, instead of the power split ratio ρ, the estimated amplitude for E_x, $\sqrt{\rho}$, is presented, since it is obtained directly from the filter h_{xx}, as shown above.



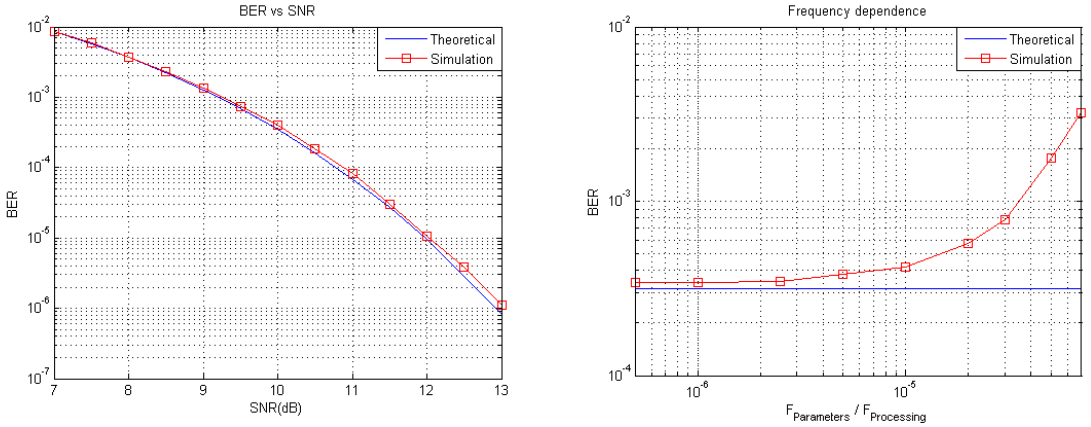
a) b)

Figure 5.4 Simulation results for a) amplitude $\sqrt{\rho}$ and b) phase difference δ

The estimated parameters follow closely the expected values. Some noise is still present in the estimations, showing that the convergence factor μ should be smaller. However, this would result in a circuit that would be able to track even smaller frequencies of variation in the SOP of the received signal.

When the power of one of the polarizations is zero, the estimated phase difference slightly deviates from the correct value since, in this case, one of the filters will be updated considering only noise. However, the estimation quickly converges to the expected values when the power returns to both signals.

Considering the same frequency of variation for the SOP (slightly under the $800Hz$), a MC analysis was performed, and the results are shown in Figure 5.5a). Also, by setting the SNR at $10dB$, different frequencies of variation for the parameters ρ and δ were used to test the response of the circuit under different conditions. The obtained results are presented in Figure 5.5b), where the considered frequencies are normalized to the clock frequency used in the equalizer (in this case, $155.52MHz$).



a) b)

Figure 5.5 a) BER vs SNR results and b) frequency-dependent performance for the CMA

In Figure 5.5a), the obtained results follow closely the expected values, proving that the circuit is operating correctly. However, in Figure 5.5b), it is possible to see that the output is quickly degraded as the frequency of variation for the input SOP increases. This can be problematic if the processing frequency is low, or if different channels are supposed to be processed by the same circuit in a sub sampling architecture (which, as will be seen in chapter 7, represents considerable hardware savings)

One of the imposed specifications is that the selected algorithm must be able to track variations in the SOP of some kHz,. This circuit is not able to ensure those requirements, and, for that reason, other solutions were studied. By analyzing the particular situation where single polarization is used, it is possible to obtain different algorithms that are able to cope with oscillations of $50kHz$ in the received SOP (the worst registered case [31]), as will be seen in the next subsections.

5.2. First implementation of Feedforward

5.2.1. The algorithms

A solution to implement a polarization tracker has been presented in [19]. This solution is given the name of Feedforward because all the data flows have the same direction. Considering the two received polarizations, one can perform the division between both polarizations

$$r(i) = \frac{E_x(iT)}{E_y(iT)}. \tag{5.6}$$

After this step a low pass filter must be applied to reduce the noise, obtaining an average of the division result, $\bar{r}(i)$. Through (5.7) the phase modulation θs is eliminated.

$$|\bar{r}(i)| = \frac{\sqrt{\rho}}{\sqrt{1-\rho}} \frac{e^{j(\theta s+\delta)}}{e^{j\theta s}} = \frac{\sqrt{\rho}}{\sqrt{1-\rho}} e^{j\delta} \tag{5.7}$$

If the power is all in one of the received polarizations (considering the case where $\rho = 1$) this implementation will have some problems since the division won't work properly. Some modifications will have to be made in the circuit to avoid this situation.

The magnitude of the filtered signal $\bar{r}(i)$ will be a function of ρ, the power splitting factor. Knowing this, one can easily determine the splitting factor through

$$\frac{|\bar{r}(i)|^2}{|\bar{r}(i)|^2+1} = \frac{\frac{\rho}{1-\rho}}{\frac{\rho}{1-\rho}+1} = \rho, \ \rho \neq 1. \tag{5.8}$$

Once again $\rho \neq 1$ is a restriction to the proper work of this solution. For $\rho \approx 0$, this implementation will also introduce error in the phase estimation, as will be seen ahead.

Knowing the power splitting factor, the missing factor is δ, the phase difference between the two received polarizations, which corresponds to the phase of the filter result. One must then extract only the phase information of $\bar{r}(i)$,

$$\delta = \angle \bar{r}(i). \tag{5.9}$$

Note that, when $\rho \approx 0$, $\bar{r}(i)$ will be close to zero and any phase information will be lost. For this reason the estimation of δ will be incorrect but, since E_x has no power for this situation ($\rho \approx 0$) there is no need to compensate a phase difference.

After having these parameters it is possible to proceed to the reconstruction of the original signal. A phase rotation in one of the received polarizations must be performed to align the phase of both signals,

$$E_x^1 = e^{-j\delta} E_x. \tag{5.10}$$

By performing this alignment, both signals and, consequently, the output will share the phase of E_y. This does not represent a problem since DQPSK is used, which removes any slow varying phase offset that may be present in E_y.

The last step is now compensating the power split. Since

$$\cos^2(\partial) + \sin^2(\partial) = \sqrt{\rho}^2 + \sqrt{1-\rho}^2 = 1, \tag{5.11}$$

the received SOP can be also characterized as

$$\begin{bmatrix} E_x \\ E_y \end{bmatrix} = \begin{bmatrix} \cos(\partial) e^{j\delta} \\ \sin(\partial) \end{bmatrix} E_s. \tag{5.12}$$

The objective now is to align both received signals and reconstruct E_s. This is done using a maximal-ratio combiner. Considering only the real component of the signal, vectors can be used to represent the received information, as seen in Figure 5.6.

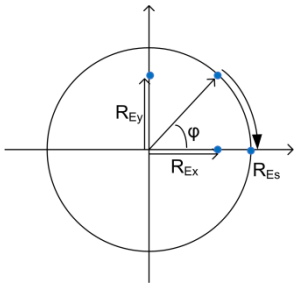


Figure 5.6 Amplitude rotation

The shown rotation can be obtained using the rotation matrix

$$\begin{bmatrix} cos(\partial) & \sin(\partial) \\ -sin(\partial) & cos(\partial) \end{bmatrix} = \begin{bmatrix} \sqrt{\rho} & \sqrt{1-\rho} \\ -\sqrt{1-\rho} & \sqrt{\rho} \end{bmatrix} \tag{5.13}$$

and, in the horizontal axis, the original signal will be reconstructed

$$E_T = \sqrt{\rho}\left(\sqrt{\rho}E_x + N_x\right) + \sqrt{1-\rho}\left(\sqrt{1-\rho}E_y + N_y\right) = E_s + \sqrt{\rho}N_x + \sqrt{1-\rho}N_y, \tag{5.14}$$

where N_x and N_y represent the noise in the two polarizations, considered to be independent and have the same power, P_N. With this in mind, the signal and the noise will have an output power of

$$E[E_s^2] = P_S \tag{5.15}$$

$$E\left[\left(\sqrt{\rho}N_x + \sqrt{1-\rho}N_y\right)^2\right] = \rho E[N_x^2] + (1-\rho)E\left[N_y^2\right] = P_N. \tag{5.16}$$

At the input the total SNR was affected by P_S and $2P_N$, and considering ideal compensation at the output the SNR is affected only by P_S and P_N, which translates in a 3dB gain in the SNR since only half of the total noise power is received.

5.2.2. **Circuit and simulation**

Having in mind the hardware implementation of this algorithm, the previous steps can be implemented using the circuit present in Figure 5.7.

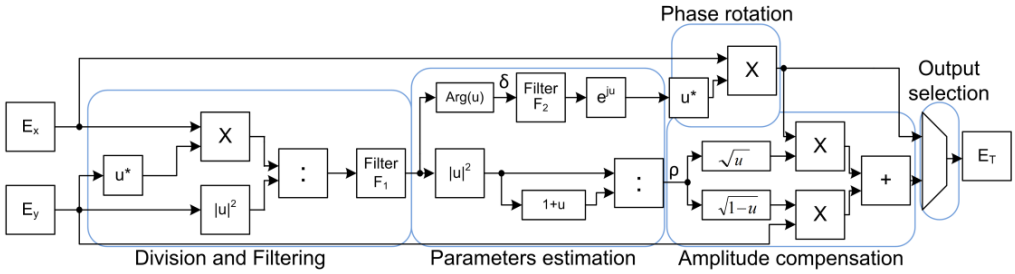


Figure 5.7 Block diagram of the Feedforward concept

The division of two complex numbers cannot be performed directly in hardware. The operation is then equivalent to

$$\frac{a+jb}{c+jd} = \frac{(a+jb)(c+jd)^*}{(c+jd)(c+jd)^*} = \frac{(a+jb)(c+jd)^*}{|c+jd|^2}. \quad \textbf{(5.17)}$$

If $|c+jd|$ is close to zero ($\rho \approx 0$), the division will be problematic.

Again thinking in the hardware implementation, infinite impulse response (IIR) filters are used, since they require less hardware than their equivalents finite impulse response (FIR). The biggest problems when using IIR filters is the non-linear phase response and possible instability due to feedback, but these will not have a big impact in the results. The used IIR filters have the following transfer function

$$H(z) = \frac{1}{2^\varepsilon} * \frac{z^{-1}}{1 - \frac{2^\varepsilon - 1}{2^\varepsilon} z^{-1}} \quad \textbf{(5.18)}$$

where the coefficients are a power of two to simplify the hardware implementation, since a division by two corresponds to a binary shift. The output can be roughly seen as the average of 2^ε inputs (in fact the older inputs have less influence in the output).

By analyzing the filter response to noise and signal, $\varepsilon = 6$ was selected to implement the filter $F1$ in this circuit. The phase and amplitude responses for this filter are presented in Figure 5.8.

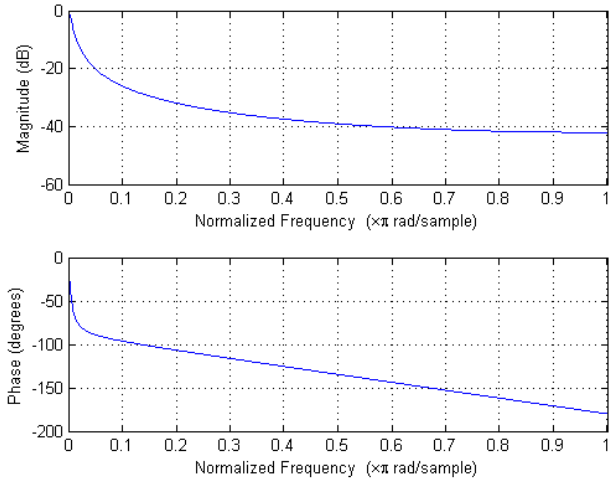


Figure 5.8 Response of the IIR filter considering $\varepsilon = 6$

The main problem of the selected IIR filter is the abrupt variation in the phase response. However, in the worst possible case, ρ and δ have a normalized frequency of $6.4 * \pi * 10^{-4} rad/sample$ and, for these values, the output phase is affected by approximately 0.122 radians, causing practically no harm to the circuit. A small attenuation of $-0.11dB$ is also introduced.

The filter has a cut frequency of $\frac{\pi}{200} rad/sample$ or $375kHz$, which is a large value but, in tests, this has proven to be enough to mitigate the noise influence in the determination of the parameters without introducing considerable attenuation or delay, which would deteriorate the circuit response.

After the filter, δ and ρ can be obtained and the circuit can reconstruct the original signal. To track the phase difference, the function $arg()$ can be directly applied. The results are then applied a low pass filter $F2$, making impossible instantaneous phase rotations. To obtain filter $F2$, $\varepsilon = 4$ was selected, resulting in

$$H(z) = \frac{1}{16} * \frac{z^{-1}}{1-\left(1-\frac{1}{16}\right)z^{-1}}. \quad \textbf{(5.19)}$$

The value returned by $arg()$ function may suffer discontinuous jumps from $-\pi$ to π. To correctly follow the estimated angle the filter needs to detect these transitions, which can be done simply by comparing the input value with the last filtered output. When a transition is detected, the value stored in the filter is multiplied by -1 before performing the loop. It is easier to see the filter and its constituent blocks in Figure 5.9, where b represents a constant small value, used to ensure that jumps caused by noise are controlled. If a FIR filter was used this task would have higher complexity.

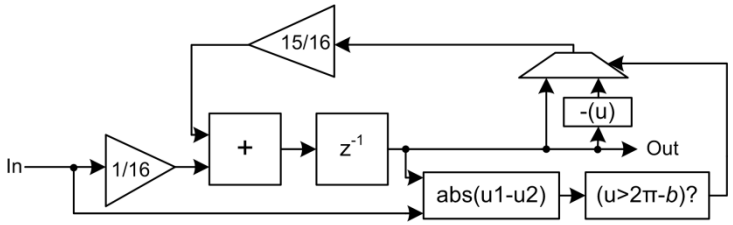
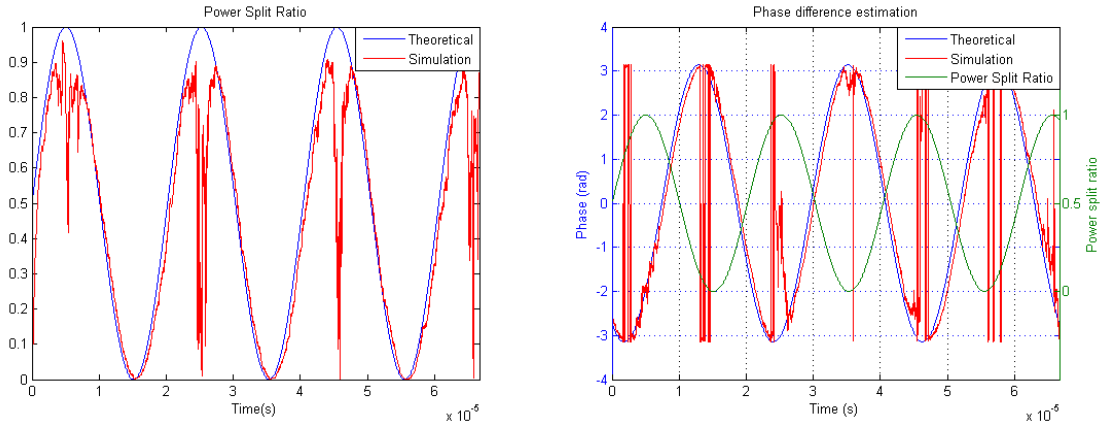


Figure 5.9 Block diagram of the IIR filter used to accommodate the phase shifts from $-\pi$ to π

When all the power is present in E_x ($\rho = 1$) the denominator of the first division is only constituted by noise, as seen before. This means that the output $r(i)$ will be unknown and, in hardware implementation, overflows may occur. This result will influence the estimated ρ and δ, as can be seen in Figure 5.10.



a) b)

Figure 5.10 Simulation results for a) ρ and b) δ using the circuit shown in Figure 5.7

In the results for the phase difference δ, the power split ratio was included to study its effects on the estimated values. The instantaneous phase jumps between -3.14 and 3.14 radians represent jumps of 2π radians and can be ignored. When ρ is close to 1, it is possible to see that both estimations fail. To correct this problem a simple solution can be implemented by stopping the normal mode of operation of the circuit. The power of E_y is constantly analyzed and, when under a determined threshold, the output is switched to use only E_x after the phase compensation, E_x^1. Also, the last known correct value of δ is preserved, by freezing the filter $F2$, to correct the signal E_x in the hope that δ does not suffer high variations when the circuit is in this alternative mode of operation. This way, the errors from the division do not affect the estimated phase. When the power returns to $|E_y|^2$ the estimated phase converges to the new correct value. In this case, to allow the stored phase, δ_{old}, to converge to the correct value without causing interference between signals, two thresholds are used to monitor the power of E_y. If both signals E_x^1 and E_y were added directly and $\delta_{old} - \delta = \pi$, destructive interference would occur.

By observing the results obtained in Figure 5.10, the estimations are incorrect for $\rho > 0.85$ (for a total SNR of $10dB$ for the input signal), meaning that the threshold can be set at $|E_y|^2 = 0.15$. Since the phase is filtered by $F2$, no instantaneous phase rotations can occur. The resulting circuit is present in Figure 5.11.

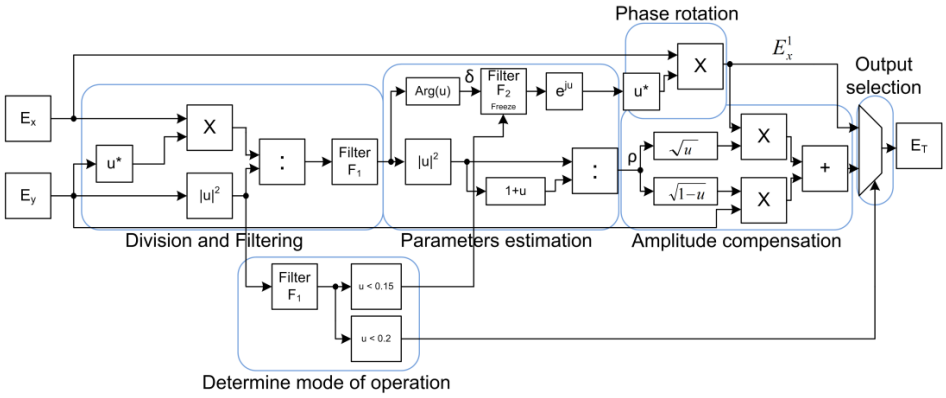


Figure 5.11 Block diagram for the first version of the Feedforward algorithm

Considering the worst case scenario, the results for the estimated phase difference δ are presented in Figure 5.12. The estimations for ρ will be the same from Figure 5.10a), since the same circuit is used for that parameter.

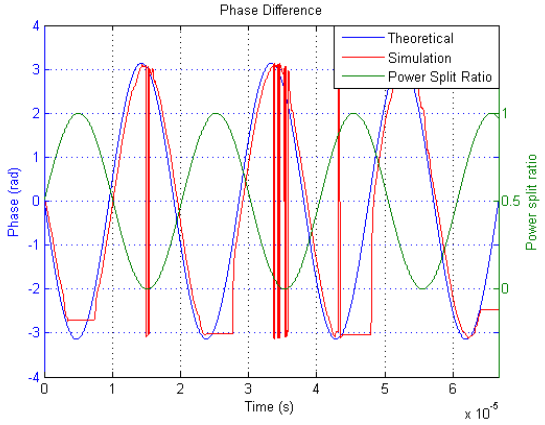
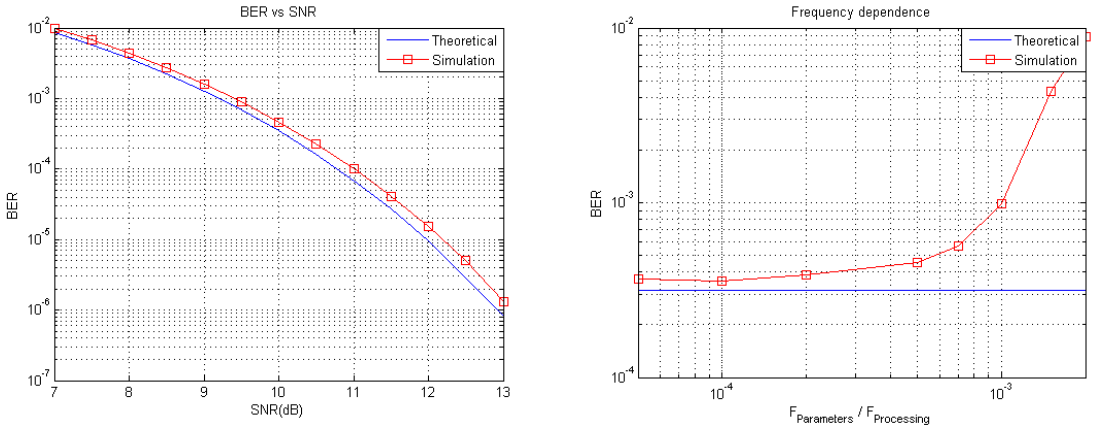


Figure 5.12 Simulation results for δ using the circuit shown in Figure 5.11

Despite the amplitude's incorrect values, these do not represent a problem now, since only E_x is used when $\rho > 0.85$. For these cases it is possible to see that the estimated phase is constant. However, when the circuit returns to the normal mode of operation, the estimated δ quickly converges to the new correct values without instant phase rotations, due to $F2$. The estimated phase being constant during the problematic values of ρ does not represent a problem since only one polarization is used in the output, and no destructive interference can occur.

In order to test the quality of the output signal, a MC simulation has been performed, considering the worst case scenario to test the ability of the circuit to follow variations of $50kHz$ in the SOP of the received signal. The results are present in Figure 5.13a). Also, after setting the input SNR at 10dB, the performance of the circuit for different frequencies of variation for the considered parameters ρ and δ was studied, and the results are presented in Figure 5.13b). Again, the frequencies were normalized considering the processing frequency for the circuit, $155.52MHz$.



a) b)

Figure 5.13 a) Simulation BER results for the worst case scenario and b) performance for different frequencies of variation using the first Feedforward approach

The results obtained in Figure 5.13a) follow closely the theoretical line even in the worst case scenario, something that was impossible to obtain with the CMA algorithm. A small SNR penalty is introduced due to the delays in the estimation of ρ and δ and to the fact that up to 15% of the total power present in the signal can be lost, when the circuit is in the alternative mode of operation.

The results from Figure 5.13b) show that the circuit is able to track normalized frequencies up to 10^{-3} ($155.52kHz$ for the considered processing rate of $155.52MHz$) before reaching the BER of 10^{-3} at a SNR of $10dB$.

This circuit is able to estimate the values ρ and δ with no problems, consisting in the first option to implement the polarization tracker. However, the problems obtained due to the division depend on the SOP of the received signal, and degrade the performance of the circuit. There is a better implementation for this algorithm, where the problems from the division are avoided, as shown in the next section.

5.3. Second implementation of Feedforward: changing divisor

5.3.1. The algorithms

In the previous implementation, when the power of E_y was under a certain threshold, the circuit would malfunction due to the fact that the divisor of the central division was only constituted by noise. By controlling the power of the divisor, this situation can be avoided. Remembering the problem, when $|E_y|^2$ is close to zero, $|E_x|^2$ is close to one. Under these conditions, this last value could then be used as the divisor for the division.

Let us consider that the performed division is

$$r_1(i) = \frac{E_x E_y^*}{|E_x|^2} = \frac{\sqrt{1-\rho}}{\sqrt{\rho}} e^{j\delta}, \quad \textbf{(5.20)}$$

where the same dividend is used from (5.6) but the divisor is now $|E_x|^2$. The phase information after the division will remain the same, since only an amplitude factor was introduced. The estimation of ρ, however, will now have to be changed to match the new situation

$$\frac{1}{1+|\overline{r_1}|^2} = \frac{1}{1+\frac{1-\rho}{\rho}} = \rho \, , \rho \neq 0. \tag{5.21}$$

Under these conditions, the problematic situation is now when the power is all present in the signal E_y ($\rho \neq 0$), the contrary from the previous case. Exploring these situations, it is possible to obtain a circuit where the divisor for the division is always the signal with the highest power, which means that the division will never have problems and the estimations can be used independently of the SOP of the received signal.

5.3.2. **Circuit and simulation**

By changing the divisor for the central division in accordance with the SOP of the received signal, it is possible to correct the previous limitations of the circuit. If the power of both polarizations is tracked, instead of only tracking $|E_y|^2$ as was done in the previous implementation, it is possible to detect if $|E_x|^2 > |E_y|^2$ (or the inverse condition) and change the divisor (and the additional logic used to estimate ρ). The new circuit is presented in Figure 5.14.

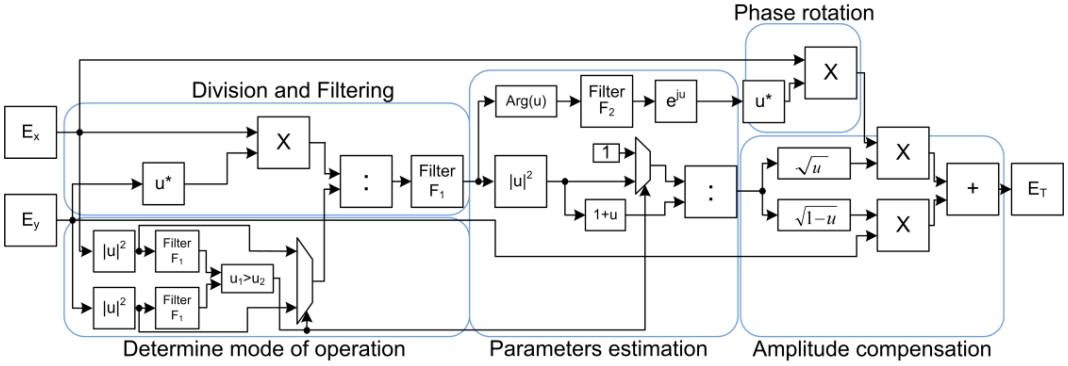
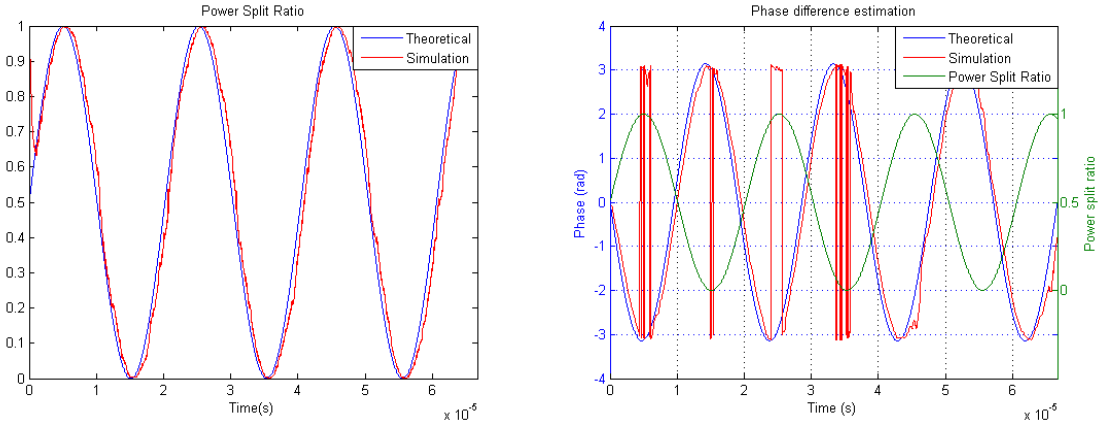


Figure 5.14 Block diagram for the second version of the Feedforward algorithm

The results obtained for the parameters estimations are presented in Figure 5.15. The circuit has the desired behavior, being able to track both parameters under any received SOP. The estimated phase difference δ slightly deviates from the correct values when the power of one of the polarizations is zero because, under these conditions, the phase returned from the $arg()$ function is only constituted by noise. Due to the used filter these values do not affect significantly the estimations and, when the power returns to both polarizations, the estimation quickly converges to the correct values.



a) b)

Figure 5.15 Simulation results for a) ρ and b) δ using the circuit shown in Figure 5.14

Again a MC simulation was performed and the results are shown in Figure 5.16. This time, no frequency dependence tests were performed because this circuit presents the same dynamics as the previous implementation, which means that the response will be the same (Figure 5.13b)).

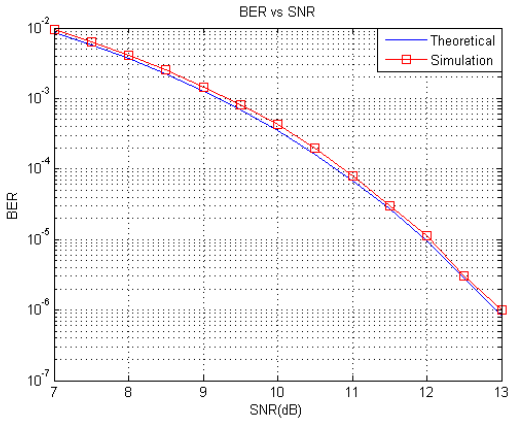


Figure 5.16 Simulation BER curve for the worst case scenario using the second Feedforward implementation

The penalty introduced due to the context switching in the first implementation of the Feedforward circuit has been eliminated, and this solution presents better overall results.

5.4. Feedback

5.4.1. The algorithms

The previous Feedforward implementations have the problem of requiring divisions which, as will be seen in chapter 6, present high hardware requirements. When considering a DOP of 8, which is the final objective, the hardware must be replicated, resulting in demanding and expensive circuits. To avoid this problem a new solution was developed, relying in feedback to reduce the hardware complexity, an idea close to the steps of the CMA algorithm.

While in the Feedforward algorithm ρ and δ were determined sharing some of the same steps, in the Feedback algorithm they require independent circuits. For this reason the different solutions will be explained separately.

Phase estimation

Feedback circuits are usual in control or analog electronics, where motors can be controlled or gain can be specified for amplifiers. There are also feedback-based algorithms used in optical communication systems. The most commons are the automatic gain control loops, used to control the gain of an amplifier prior to reception or to normalize amplitudes, as seen in the CMA algorithm. Other examples are the PLLs, used to tune local oscillators [9], where a value proportional to the phase error is applied to a VCO, in electrical PLL (EPLL), or to the drive current of a local oscillator, in optical PLL (OPLL).

Here, an algorithm with the same principles of an EPLL is implemented, where the compensation is performed completely in the digital domain. The suggested implementation is presented in Figure 5.17.

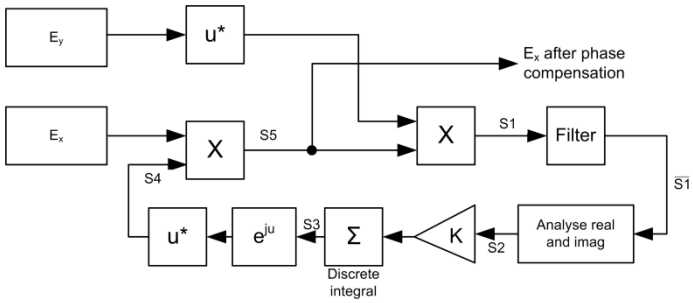


Figure 5.17 Digital phase locked loop

To simplify the comprehension of the circuit let us consider, for the moment, that both input signals E_x and E_y have unitary amplitude, $e^{j(\theta s+x)}$ and $e^{j(\theta s+y)}$. Let us also ignore, for the moment, the first multiplication of E_x by $S4$ (in fact, at the beginning $S4 = 1$). With this in mind it is easy to see that the signal $S1$ consists only in the phase difference between both channels, removing the phase modulation θs: $S1 = e^{j(x-y)}$. The goal of the circuit will be obtaining $S1 = 1$, situation where the phase difference is zero, which means that $S5$ is in phase with E_y. After removing the phase modulation, $S1$ must be filtered, $\overline{S1}$, to eliminate some of the noise.

The key block of this circuit is “Analyze Real and Imag”. By analyzing both the real and the imaginary components of $\overline{S1}$ it is possible to determine in which quadrant it is located. If it is located in the 1st or 2nd quadrants (including the negative real axis), the circuit should perform a clockwise rotation of $\overline{S1}$, since this is the shortest “rotation path”. If, in opposite, $\overline{S1}$ is located in the 3rd or 4th quadrants, the circuit should perform a counter-clockwise rotation. This concept is shown in Figure 5.18.

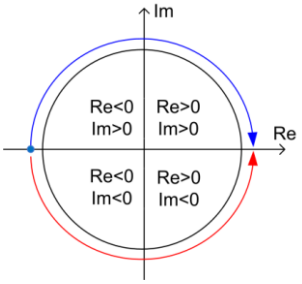


Figure 5.18 Desired rotations

If the desired rotation is to be performed clockwise, a positive number is introduced in the discrete integral. The gain K introduced is a positive small value to ensure stability and smooth rotations. As a result, $S3$ will increase, $S4 = e^{-jS3}$ and $S5 = e^{j(\theta s + x - S3)}$, which means that the signal $S1$ will become $S1 = e^{j(x-y-S3)}$. Since $S3$ is a rising value, the phase of $S1$ is decreasing, and the rotation is being performed clockwise, as desired. If, on the contrary, the desired rotation is to be performed counter-clockwise, a negative number will be introduced in the integral. The resulting $S3$ will be decreasing and, following the same steps, it is easy to conclude that the rotation will be performed counter-clockwise, as desired.

The same result is obtained simply by introducing the imaginary component of $\overline{S1}$ (except when $\overline{S1} = -1$) in the discrete integral, since $Imag_{S1} > 0$ leads to a clockwise rotation and $Imag_{S1} < 0$ leads to a counter-clockwise rotation. The only problem would be $\overline{S1} = -1$. In fact, when the real component of $\overline{S1}$ is negative a big phase difference must be compensated. However, the more negative this real component is, the lower the imaginary component will be and the resulting rotation will be slow, which may cause a destructive interference between polarizations. One could then use a different approach in this critical case: if the real component of $\overline{S1}$ is smaller than a certain threshold, a large value with the same signal as the imaginary component of $\overline{S1}$ is introduced in the integrator, forcing a fast rotation in the desired direction. This concept is shown in Figure 5.19.

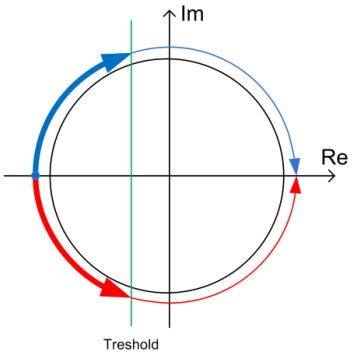


Figure 5.19 Desired rotation with a threshold. Thicker arrows represent faster rotations

After some trials, a threshold of $Real_{\overline{S1}} < -0.2$ was selected since, in some cases, noise could lead the real component of $\overline{S1}$ to be negative. Also, $K = 0.0156$ was selected to the feedback loop and $\varepsilon = 4$ was selected to dimension the low pass IIR filter, presenting a good noise and speed response without affecting the stability of the feedback loop.

Remembering the received polarizations (3.1), the inputs are not normalized, which will lead to smaller components in $\overline{S1}$ and result in slow rotations. This can be corrected by introducing a gain before the filter, so that the real and imaginary components have a considerable weigh when analyzed. After some tests a gain of 9 was chosen.

When the power is completely in one of the polarizations ($\rho = 0$ or $\rho = 1$) $S1$ will become zero and the phase estimation will be determined only by the noise present in both polarizations, which may cause a slow drift in the estimated value. This situation does not represent a problem though since, in this situation, only one signal has power and no destructive addition will occur.

The simulation results are present in Figure 5.20.

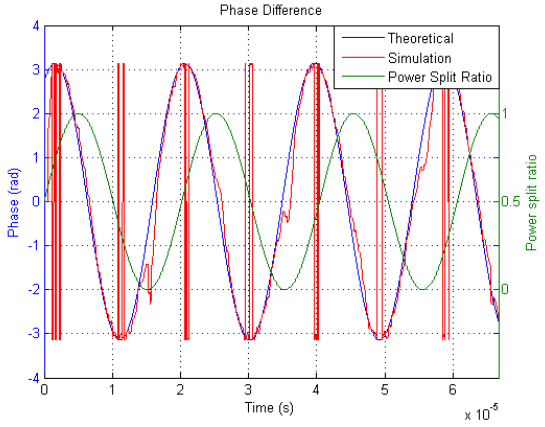


Figure 5.20 Phase estimation using the circuit in Figure 5.17

The estimated phase follows closely the expected values except when $\rho = 0$ or $\rho = 1$. In these situations the estimated values deviate randomly from the correct values but, when the power returns to both polarizations, they rapidly converge to the correct values, showing that the circuit is working properly.

Amplitude estimation

The amplitude compensation can be performed independently for each polarization. Considering one of the polarizations $\sqrt{\rho}e^{j(\theta s+\delta)}$, the value of $\sqrt{\rho}$, used to compensate the amplitude of the received signal, can be simply determined by calculating the power of the input and then the square root of that result

$$\sqrt{\left|\sqrt{\rho}e^{j(\theta s+\delta)}\right|^2} = \sqrt{\rho}. \quad \textbf{(5.22)}$$

The amplitude correction can be performed using the circuit presented in Figure 5.21.

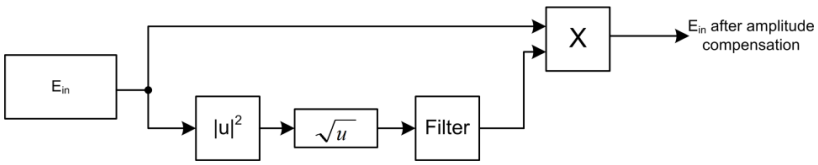


Figure 5.21 Circuit used to estimate and compensate amplitude using a square root

A filter can be applied to the estimated $\sqrt{\rho}$ to reduce noise. Once again, an IIR filter is used. With the hardware implementation in mind, the same circuit can be simplified and implemented using a feedback loop to estimate the square root. The obtained circuit is presented in Figure 5.22.

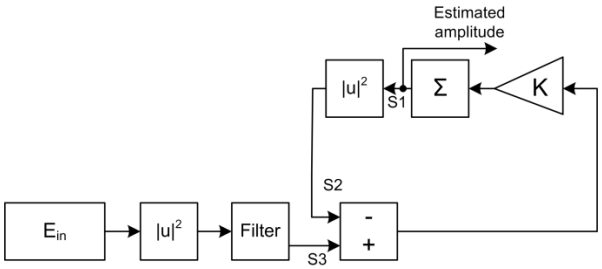
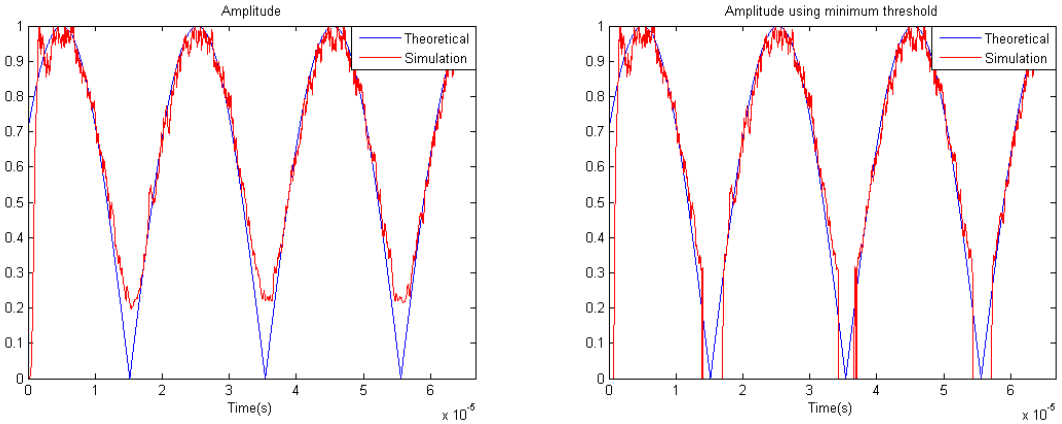


Figure 5.22 Circuit used to estimate and compensate amplitude using feedback

$S3$ is the power of E_{in}. Considering that $S2 < S3$, the stored value $S1$ will increase and cause $S2$ to increase until $S2 = S3$. In this situation, $S1 = \sqrt{S3}$, the amplitude of the input signal. This method used to estimate the amplitude suffers from one problem: when the amplitude is zero, the estimated power will represent the power of the noise component, which will create a minimum floor for the estimated amplitude. This can be corrected by considering a minimum threshold. When the amplitude is under a certain level, the polarization in question is neglected, similar to what was done in the first Feedforward implementation. Through some trials, an inferior threshold of 0.3 (10% of the total power) has proven to be enough. Also, $\varepsilon = 4$ was selected to implement the IIR filter and a gain of $k = 0.02$ was used.

Two of these circuits must be used in parallel, one for each polarization. The results for the estimated amplitude are presented in Figure 5.23, considering the worst case scenario. In this case, since $\sqrt{\rho}$ is directly estimated for E_x, the amplitude of the signal is directly presented instead of ρ.



a) b)

Figure 5.23 Amplitude estimation a) without and b) with minimum threshold

The obtained results from Figure 5.23 are close to the expected values. Some of the power is lost due to the threshold but this measure prevents additional noise at the output.

Automatic gain control

The previous circuit, used to estimate the amplitude, suffers from one problem: when the input signals suffer from a gain factor, g (common to both polarizations) the estimated amplitude will be proportional to that factor (5.23). This can occur, for example, at the ADCs, where the gain introduced in the analogical signal is controlled to adapt the maximum signal amplitude to the dynamic range of the ADC. However, if the signal suffers from high noise or interferences, this maximum will be determined by the noise and not by the signal amplitude, which will result in a small scaling factor in the total amplitude for the signal. This gain can be problematic if g is a small value, resulting in further attenuation for the output signal, since the amplitude estimation becomes

$$\sqrt{\left|g\sqrt{\rho}e^{j(\theta s+\delta)}\right|^2} = g\sqrt{\rho}. \tag{5.23}$$

To correct this problem, the input power must be normalized before proceeding with the amplitudes estimation. This can be performed with the circuit presented in Figure 5.24, where the concept is similar to the circuit used in [53] for normalization of electrical signals, or to the idea used to normalize the output signal in the CMA algorithm.

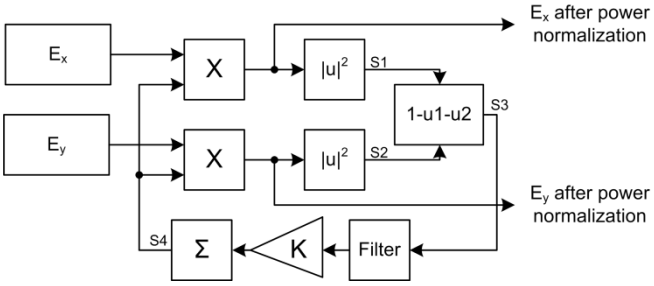


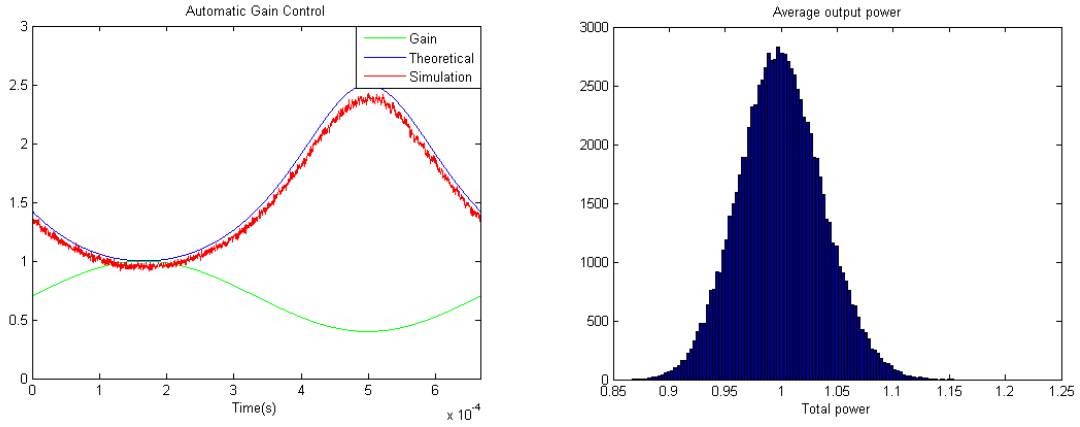
Figure 5.24 Block diagram for the automatic gain control

Considering that, initially, $S4 = 1$, $S1$ and $S2$ represent the powers of E_x and E_y, respectively. If both polarizations are affected by a gain g, the signals will be

$$S3 = 1 - (S1 + S2) = 1 - g^2\big(\rho + (1-\rho)\big) = 1 - g^2. \tag{5.24}$$

Considering that the input signals are attenuated ($g < 1$), $S3 > 0$. After some filtering, to eliminate some noise, and a small gain $K > 0$, a positive value will enter in the discrete integral, increasing $S4$. The value of $S4$ will grow until $S3 = 0$, situation where the $S4 = g_{agc} = 1/g$ and the gain factor g is removed. If, by the contrary, $g > 1$, the same steps will show that $S4$ will decrease to compensate this value.

After some tests, the feedback gain $K = 0.01$ is used and the IIR filter was dimensioned using $\varepsilon = 6$. The factor g was simulated using a sinusoid with a small oscillation frequency to test the adaptability of the circuit. The obtained results are presented in Figure 5.25. The compensation value must be the inverse of the applied gain, so the inverse of g is also represented.



a) b)

Figure 5.25 Results obtained using the circuit in Figure 5.24 for a) estimated power compensation and b) histogram of the output power

The estimated values are slightly deviated from the ideal. In the worst case the error in the estimation is close to 0.1. This happens because the noise also contributes to the estimated power in both polarizations, which will lead to an increase in the total power seen by the circuit, resulting in a smaller equivalent gain. Despite this, the total output power is always close to one, as seen in the histogram, demonstrating the good overall behavior of the circuit.

5.4.2. **Circuit and simulation**

By combining all the circuits presented before the final circuit for the Feedback algorithm is shown in Figure 5.26.

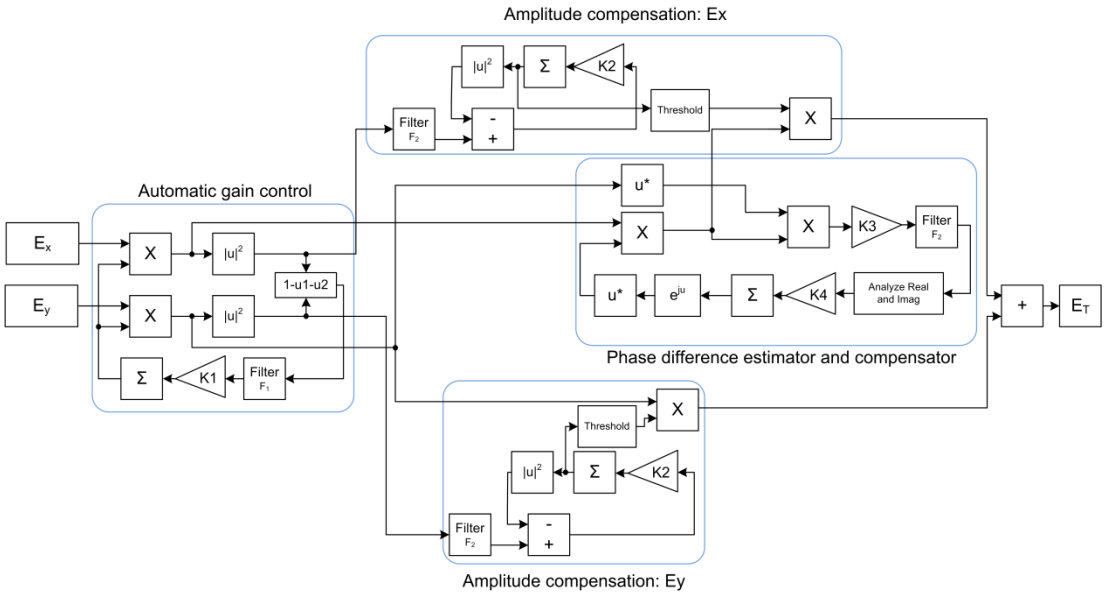
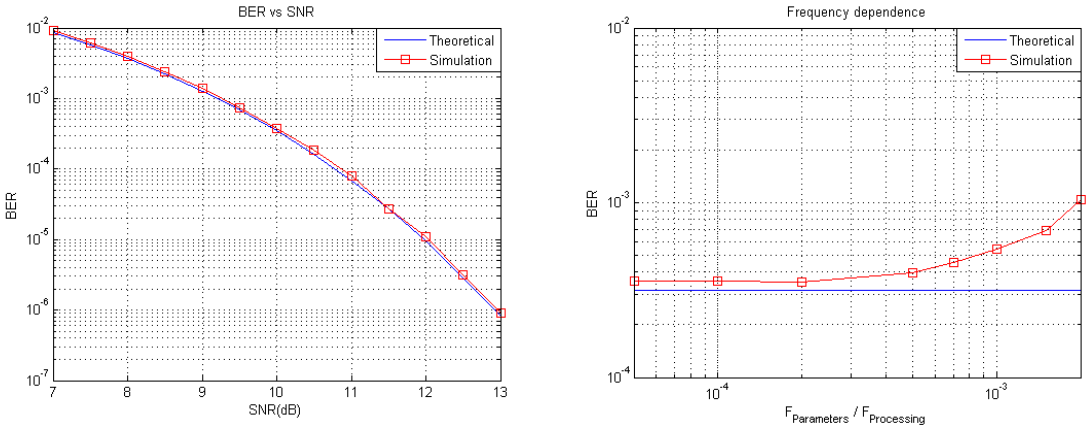


Figure 5.26 Block diagram for the complete Feedback circuit

A MC simulation was performed and the results are presented in Figure 5.27a). The frequency tests were also performed and the obtained results are shown in Figure 5.27b).



a) b)

Figure 5.27 a) BER vs SNR curve for the worst case scenario and b) frequency dependence test for the Feedback algorithm

In Figure 5.27a) the obtained BER is very close to the ideal limit showing that the compensation performed is almost perfect, even for the worst case scenario. Also from Figure 5.27b), it is possible to see that the circuit is able to track high frequency variations, up to a normalized frequency of $2 * 10^{-3}$ (or higher than $300kHz$ for the considered setup) without crossing the $BER = 10^{-3}$ limit when $SNR = 10dB$ is considered.

5.5. Conclusions

A direct comparison between the Feedforward and the Feedback implementations is performed in Figure 5.28, where the worst case scenario is considered. In this case, the CMA results are not present, since this algorithm was not able to equalize the signal for such frequencies of variation in the parameters α and δ. It is possible to see that the second Feedforward implementation and the Feedback implementation are the best algorithms to solve the problem in hands, since the SNR penalty introduced is practically inexistent. The first Feedforward implementation is the one that presents the worst results.

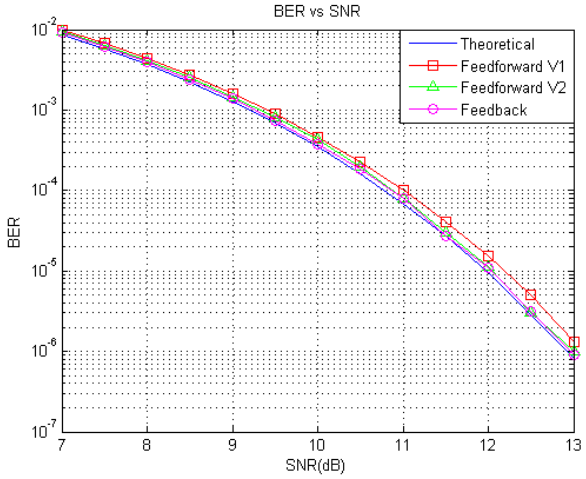


Figure 5.28 Comparison of BER results for the different algorithms in the worst case scenario

6. Hardware implementation

The algorithms were all presented sequentially in the previous chapter. However they were developed and implemented in hardware one at a time. In fact, despite starting with the presentation of the CMA algorithm in the previous chapter, this algorithm was the last to be implemented and tested in hardware. This way, the different algorithms were implemented, starting with the Feedforward solutions, going through the Feedback implementation and ending with the CMA algorithm, in an attempt to obtain a compromise between the performance and hardware requirement for each implementation. For this reason the hardware implementation and specifications for the different approaches will be presented.

6.1. First implementation of Feedforward

This first implementation of the Feedforward algorithm represented a case study for VHDL while trying to find a working solution for the problem in hands, so no hardware simplifications were made. Even if simplifications were performed the result would not change significantly, as will be seen in the next subchapters.

Before implementing the circuits, some decisions have to be taken. In Figure 5.11 it's possible to identify the blocks that constitute the circuit. To all the communications between those blocks the same data format was selected to avoid mismatches. Since the input data is in the format Q.7 (8-bit 2's complement with 7 fractional bits), it was decided that the internal signals could be represented in the format Q4.11 (16-bit 2's complement value with 4 integer and 11 fractional bits), allowing the intermediate operations to have a precision smaller than $0.5 * 10^{-3}$ and absolute amplitudes close to 17, enough to accommodate any signal that goes through the circuit without noticeable data loss.

Whenever possible, IPCores were used to implement the used components since they result in optimized implementations of the desired functions, offering better performance and / or occupying less FPGA space.

Although the Q4.11 was adopted as the signal format between blocks, some of the blocks must have different data formats in their interior for the circuit to work properly. For example, when a number is multiplied by a small gain before entering a discrete integral, in the integral the data should be represented using more fractional bits so that no major data is lost. Also, some of the IPCores do not provide directly the Q4.11 format, which means that a "shell" must be used to convert the inputs and the outputs to the desired format. Additionally, the majority of the represented signals are complex, which implicates the duplication of the buses and following hardware. The complex values are just a mere representation, as the real and imaginary components are carried by two different buses.

The basic hardware blocks are explained in Annex 1.

6.1.1. Complete circuit and hardware requirements

The hardware implementation is similar to the circuit shown in Figure 5.11. However, the hardware imposes cycle delays in most of the operations and, to ensure the

correct behavior, the pipeline must be preserved. After finding the critical path, the one where the signals present the highest delay, the other paths must be corrected introducing delays so that all the signals arrive at the same time to the output. The corrections performed in this case are presented in Figure 6.1, where the green arrows show what the delays are compensating and the blue blocks and buses indicate the presence of complex signals, which may require some sort of hardware replication.

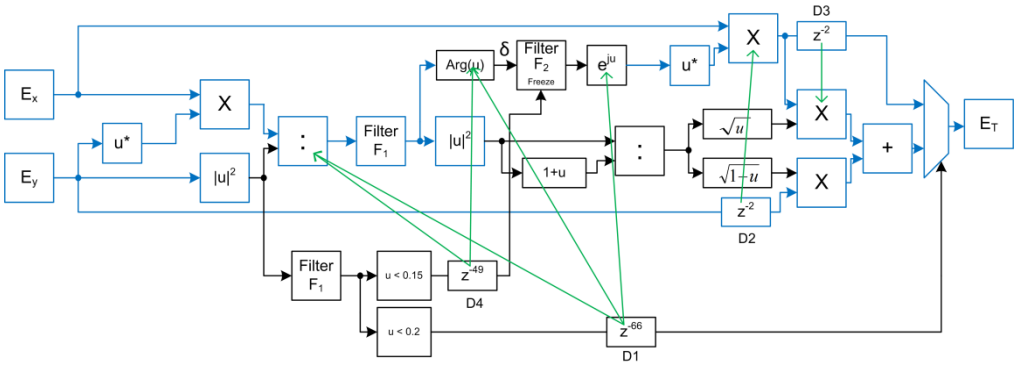


Figure 6.1 Hardware block diagram for the first Feedforward implementation

The delay $D1$ ensures that the output uses both polarizations until the incorrect results of the division affect the end of the pipeline. $D2$ is used so that both E_x and E_y have the same delay from the input to the output of the circuit. $D3$ makes sure that, even when the circuit changes its mode of operation, the samples arrive with the same delay to the output. Finally, delay $D4$ must be introduced so that the filter $F2$ processes all the correct samples introduced in the pipeline before the problems in the division are received. The circuit introduces a delay of 4 clock cycles in the processed samples, resulting from the phase (2) and amplitude (2) compensation.

In terms of hardware requirements, the Feedforward is a demanding solution. The use of the primary blocks can be seen in Table 6.1.

	Number	Primary blocks				
		Multipliers	Dividers	Square root Cordic	Sin & Cos Cordic	Atan Cordic
Complex multiplier	2	6	0	0	0	0
Complex – Real multiplier	2	4	0	0	0	0
|Complex|^2	2	4	0	0	0	0
Square root	2	0	0	2	0	0
Divisions (Complex)	1	0	2	0	0	0
Divisions (Real)	1	0	1	0	0	0
Sin & Cos	1	0	0	0	1	0
Atan	1	0	0	0	0	1

Table 6.1 Basic blocks used in the implementation of the first Feedforward algorithm

The total hardware requirements of the Feedforward implementation are present in Table 6.2, where the RS232 interface, state machine and memories were removed. The percentage values are obtained considering the resources available in the Virtex 6 FPGA, presented in Table 4.1.

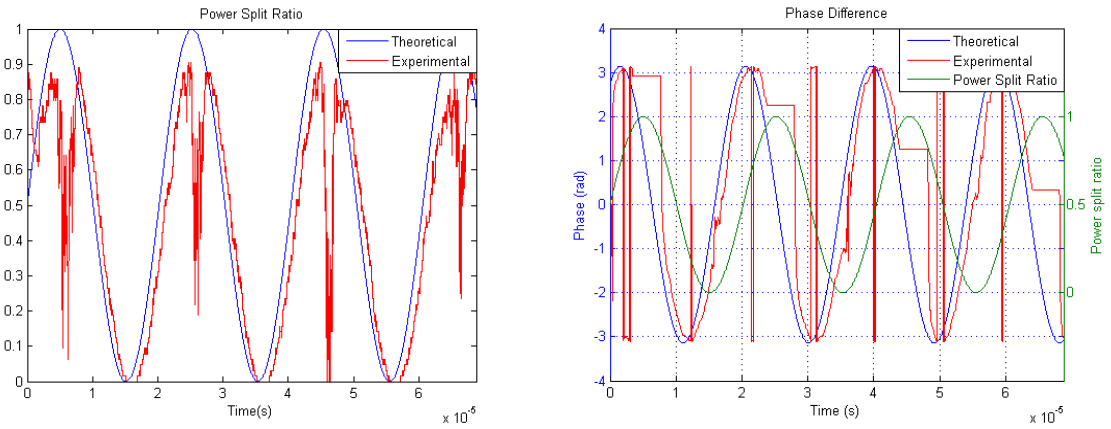
	Slice LUTs	Slice Registers	Slices	DSP48E1s
Total	6203	7223	2285	14
Total (%)	4.12%	2.40%	6.06%	1.82%

Table 6.2 Total hardware usage for the first Feedforward implementation

Some of the resources could be predicted, such as the number of DSP48E1s used, since they are only required by the multipliers. One could have an idea of the magnitude of the required hardware considering the isolated blocks, but there are elements (the filters or the adders) that weren't considered and that contribute to the total hardware. There are also cases where a slice is used only as a path between components, and is considered by the synthesis tool as a used slice. Nevertheless, more than 6% of the slices were used by the circuit which, considering the high available resources in the FPGA, shows the high resource consumption of this solution. In case of replication this could be problematic. No hardware simplification was performed in this solution. However the divisions could not be simplified and they require most of the used resources.

6.1.2. **Experimental results**

The estimations for the parameters, obtained using the circuit shown in Figure 6.1, are presented in Figure 6.2.

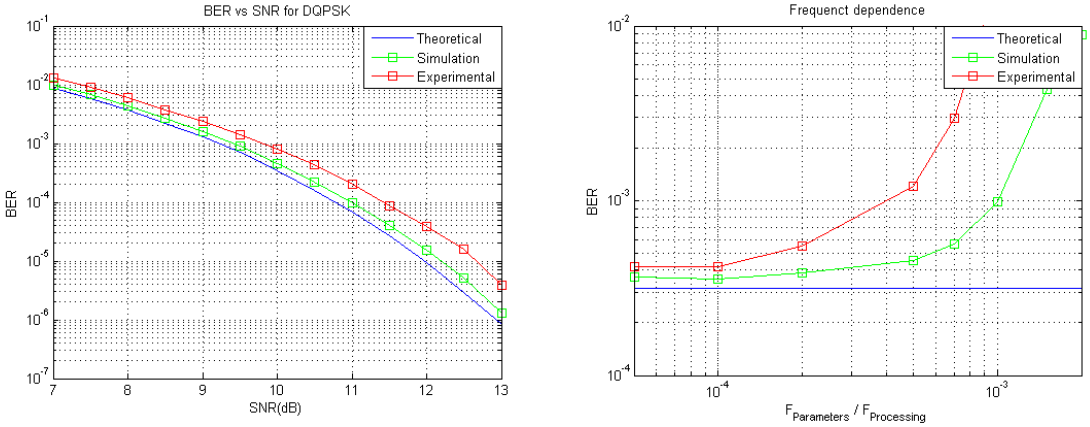


a) b)

Figure 6.2 Experimental results for a) ρ and b) δ using the circuit shown in Figure 6.1

The obtained results are consistent with the ones presented in Figure 5.12. Note, however, that the estimated components now present a considerable delay. Besides the high hardware requirements, the implicit delay is the second problem for the use of divisions. The delay introduced in both components is close to $0.4rad$ (considering the worst case scenario). This delay alone is expected to give rise to a penalty close to $0.5dB$ in the output SNR, according to Figure 4.7.

A MC analysis was performed and the results are presented in Figure 6.3. Also, after fixing the SNR of the input signal at $10dB$, different frequencies of variation for the SOP were considered, and the obtained results are presented in Figure 6.3b).



a) b)

Figure 6.3 Experimental BER results for a) the worst case scenario using the first Feedforward implementation and b) frequency dependence of the circuit

The results presented in Figure 6.3a) are consistent with the ones obtained in the simulations performed. However, the SNR penalty experienced in the experimental results is higher, close to $0.7dB$. This value is not far from the expected due to the delay in the estimated parameters, showing that other factors degrade the output, such as the threshold applied to the amplitude or other causes related with the hardware, like the quantification error. In Figure 6.3b) it is possible to see that the output of the circuit degrades quickly as the frequency of variation of the received SOP increases. It is possible to see that, for the worst case scenario, the BER is the double of the ideal results, which is confirmed in Figure 6.3a). This fast degradation results from the use of filters and from the delay introduced by the nonlinear functions, mainly the divisions.

6.2. Second implementation of Feedforward: changing divisor

6.2.1. Complete circuit and hardware requirements

Again, the hardware delays must be introduced to preserve the pipelining. Using the same steps followed before, the resulting circuit is shown in Figure 6.4, using the same color code as in Figure 6.1.

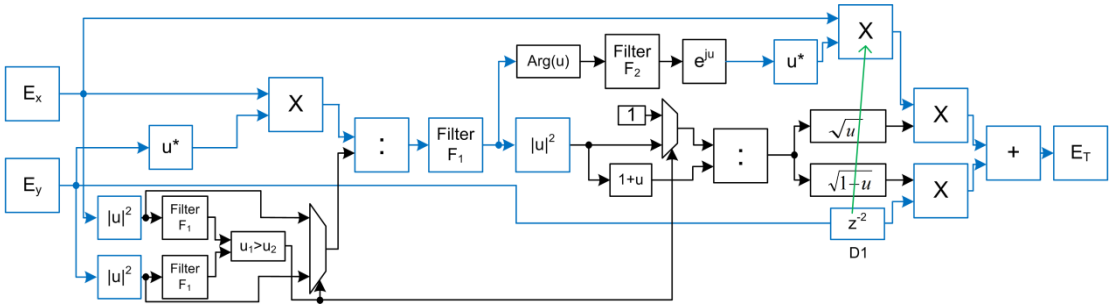


Figure 6.4 Hardware block diagram for the second Feedforward implementation

In this case, only one delay, $D1$, is required to ensure that the circuit presents the same delay to both polarizations. Again, a total delay of 4 clock cycles is introduced by the circuit.

The primary blocks used are presented in Table 6.3.

	Number	Primary blocks				
		Multipliers	Dividers	Square root Cordic	Sin & Cos Cordic	Atan Cordic
Complex multiplier	2	6	0	0	0	0
Complex – Real multiplier	2	4	0	0	0	0
$\|Complex\|^2$	3	6	0	0	0	0
Square root	2	0	0	2	0	0
Divisions (Complex)	1	0	2	0	0	0
Divisions (Real)	1	0	1	0	0	0
Sin & Cos	1	0	0	0	1	0
Atan	1	0	0	0	0	1

Table 6.3 Basic blocks used in the implementation of the second Feedforward algorithm

When compared with the previous implementation, this solution requires few more resources, to perform the commutation and estimate the power of both input signals. The results are shown in Table 6.4.

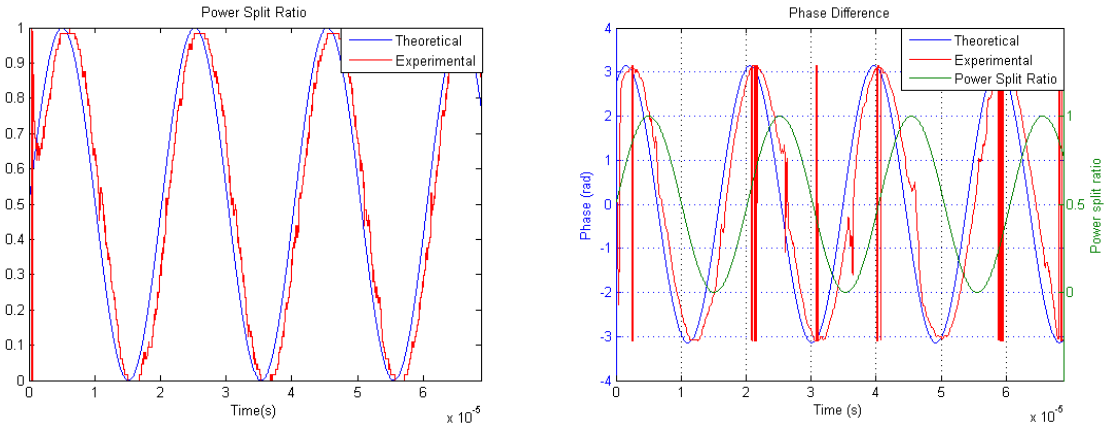
	Slice LUTs	Slice Registers	Slices	DSP48E1s
Total	6255	7248	2611	16
Total (%)	4.15%	2.40%	6.93%	2.08%

Table 6.4 Total hardware requirements for the second Feedforward implementation

A small increase in the required hardware has taken place, both in the required Slices, which represent now an additional 1% of the total available Slices, and in the number of DSP slices, due to the multipliers to estimate the second input power. However, the results are slightly better, as seen in the next section.

6.2.2. **Experimental results**

The estimations for ρ and δ are presented in Figure 6.5.



a) b)

Figure 6.5 Experimental results for a) ρ and b) δ using the circuit shown in Figure 6.4

These results show that the hardware presents the desired behavior. The amplitude is now correctly estimated independently of the SOP of the received signal. There are slight deviations from the correct values when the circuit transits between modes of operation ($\rho = 0.5$). The phase estimations are also close to the expected values. Note that when one of the polarizations has no power ($\rho = 0$ or $\rho = 1$), the estimated results deviate from the correct values, but this situation does not represent a problem since no destructive addition can occur. Again, due to the delay introduced in the estimation of ρ and δ, a SNR penalty of $0.5dB$ Is expected in the MC analysis performed. The results from such analysis are shown in Figure 6.6.

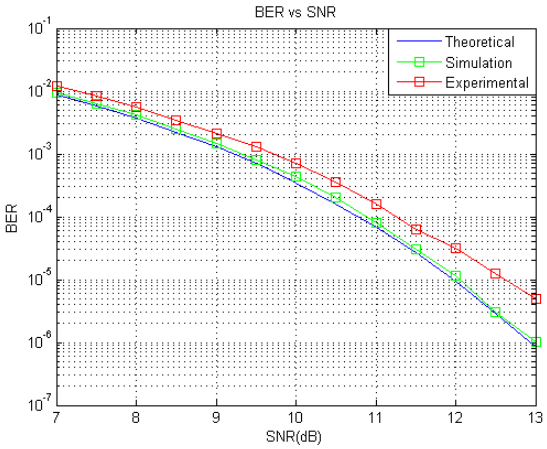


Figure 6.6 Experimental BER results for the second Feedforward algorithm in the worst case scenario

The total SNR penalty present in the results has diminished approximately $0.1dB$ when comparing these results with Figure 6.3, arising from the fact that now no amplitude truncation occurs. This value is closer to the expected penalty only due to the delay in the estimated parameters. Again, in this case no frequency tests were performed since the circuit presents the same dynamics as the previous implementation in terms of filtering and delays, meaning that the results are similar to the ones presented in Figure 6.3b).In practical situations, where ρ and δ are expected to vary slowly, in a matter of some kHz, the penalty decreases, as seen in the same figure.

6.3. Feedback

6.3.1. Complete circuit and hardware requirements

The hardware implementation of the feedback algorithm follows closely the circuit presented in Figure 5.26. Again, aiming to preserve the pipelining, delays must be introduced. However, as happened in the second Feedforward implementation, only one delay must be introduced, as shown in Figure 6.7, using the same color representation as before.

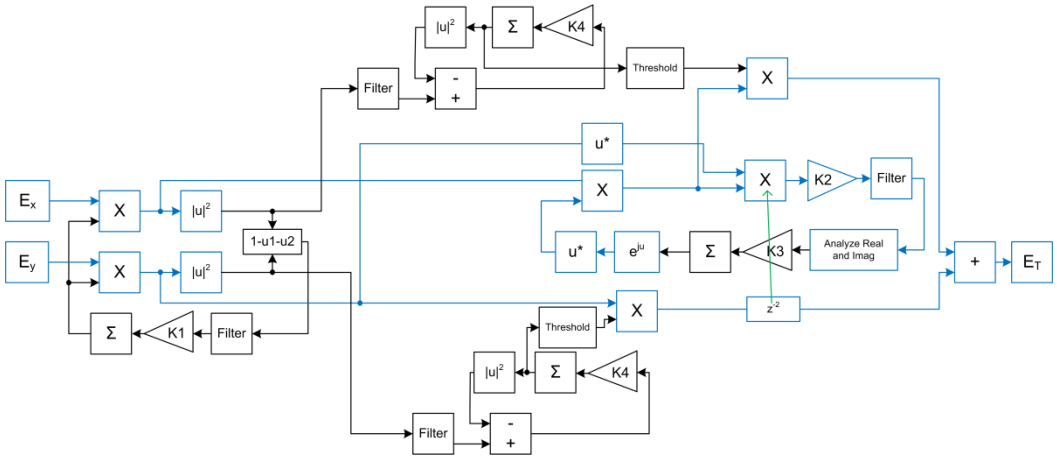


Figure 6.7 Block diagram for the hardware implementation of the Feedback algorithm

The delay is introduced to ensure that the samples of E_x and E_y arrive at the same time to the output. Six delays are introduced in the samples by the circuit, considering the automatic gain control (2), phase (2) and amplitude (2) compensation.

The circuit is completely based in products and in the e^{ju} operation. In fact, the hardware reduction obtained using the Feedback is so substantial that another reduction can be performed if the e^{ju} operation is implemented using a LUT, as described in Annex 1. The primary blocks used are presented in Table 6.5.

	Number	Primary blocks	
		Multipliers	Sin & Cos LUT
Complex multiplier	2	6	0
Complex – Real multiplier	4	8	0
$\|Complex\|^2$	2	4	0
$\|Real\|^2$	2	2	0
Sin & Cos	1	0	1

Table 6.5 Basic blocks used in the implementation of the Feedback algorithm

The total hardware requirements are shown in Table 6.6.

	Slice LUTs	Slice Registers	Slices	DSP48E1s	RAMB36E1
Total	667	252	223	20	1
Total (%)	0.22%	0.08%	0.59%	2.60%	6.25%* / 0.24%**

Table 6.6 Total hardware requirements for the Feedback implementation (*when relative to the number of RAMB36E1 and ** if relative to the total available memory)

It is possible to conclude that the implementation using the Feedback approach presents high reduction in the used resources, as expected. Only the number of multipliers and memories used has increased for a small amount. Since the Virtex6 is a DSP FPGA, with a high number of multipliers and memories, the Feedback implementation is the best solution in terms of required resources, when compared with the Feedforward algorithms.

6.3.2. **Experimental results**

The different blocks developed in simulation were tested individually before studying the total Feedback implementation. The results for the different circuits are shown in Figure 6.8.

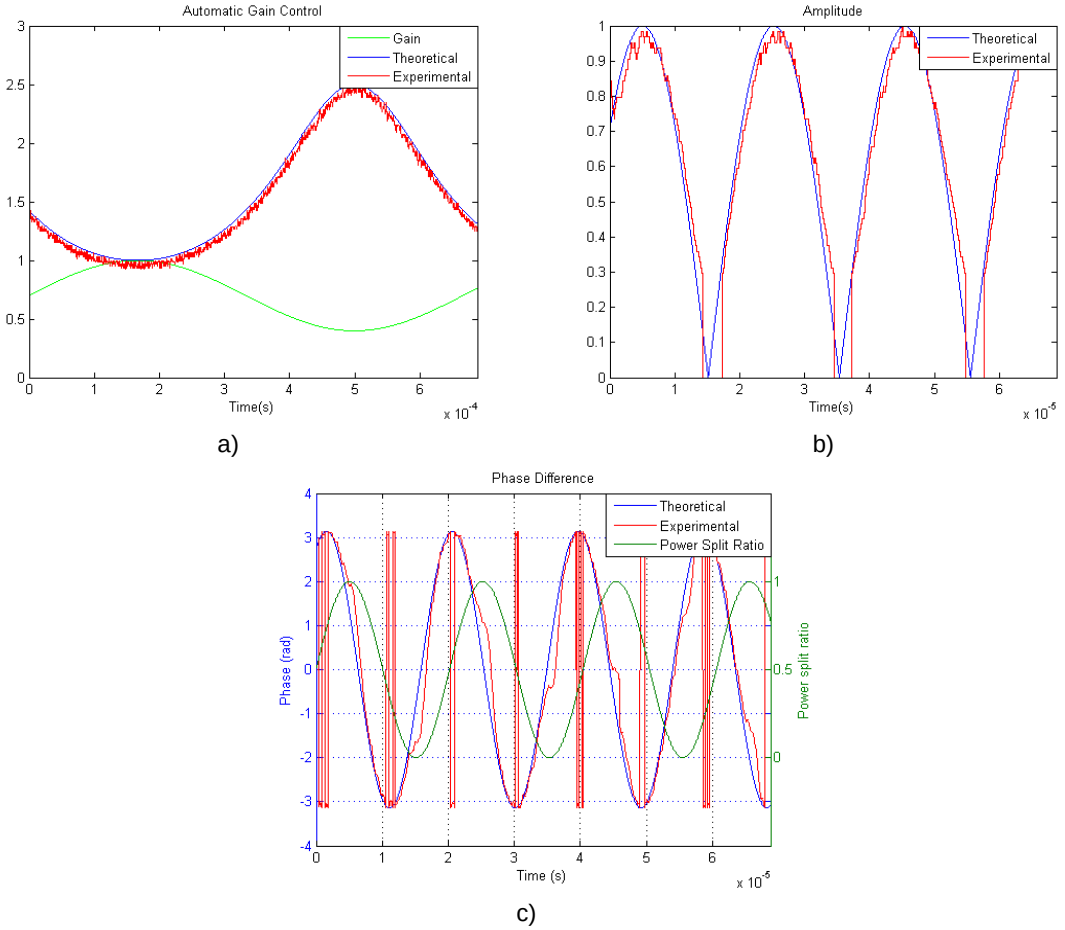
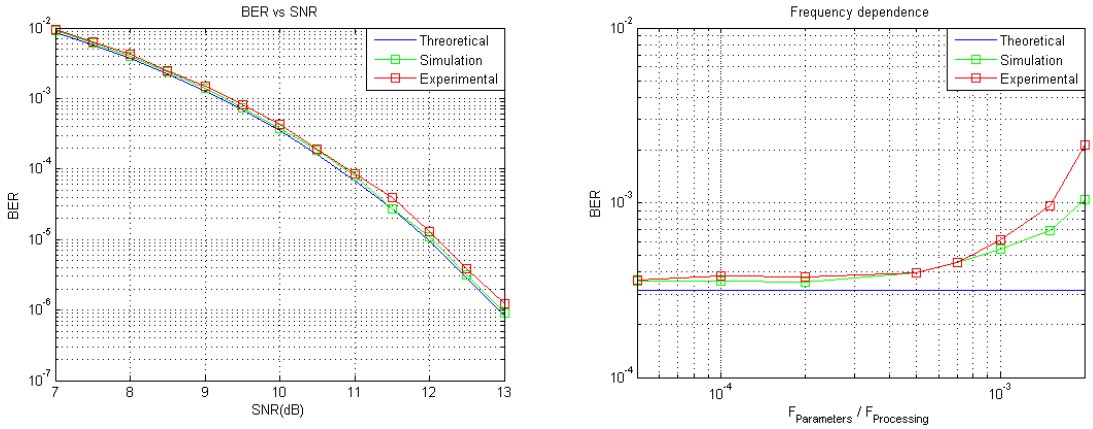


Figure 6.8 Experimental results for a) the input power normalization, b) amplitude ($\sqrt{\rho}$) estimation and c) phase difference (δ) estimation

The obtained results are consistent with the ones obtained in the simulations. Also, note that estimated parameters present a smaller delay than the results obtained with the previous Feedforward algorithms, meaning that, for the worst case scenario, this circuit will introduce a smaller penalty. A MC analysis was performed and the results are presented in Figure 6.9a). The results from the frequency dependence tests are presented in Figure 6.9b).

Note that, even with the hardware implementation, the results presented in Figure 6.9a) follow closely theoretical limit which shows that the equalization performance is almost perfect, even in the worst case scenario. Also, from Figure 6.9b), the experimental results follow closely the expected values obtained through simulation, showing that the hardware implementation does not introduce considerable delays in the estimated parameters when compared with the simulated circuit.



a) b)

Figure 6.9 a) Experimental BER results for the worst case scenario and b) frequency dependence with the Feedback algorithm

6.4. CMA algorithm

Despite not being able to track fast variations in the received SOP, it would be interesting to study the hardware implementation for the CMA algorithm, to obtain some information about its requirements.

6.4.1. Complete circuit and hardware requirements

Much like the previous examples, the hardware implementation of the CMA algorithm follows closely the block diagram circuit presented in simulation, Figure 5.3. However, delays must be introduced to ensure the pipeline and compensate for hardware-induced delays. With this in mind, the hardware implementation is represented in Figure 6.10, again with the same color code.

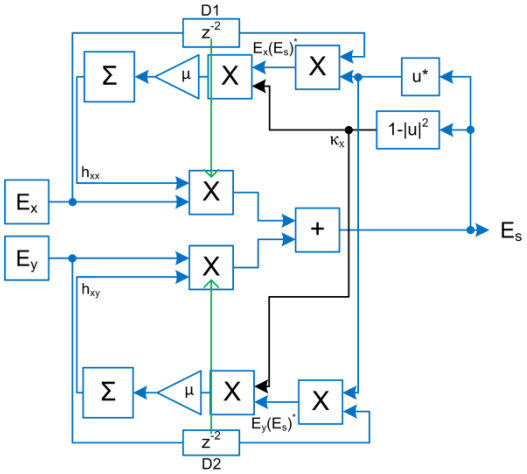


Figure 6.10 Block diagram for hardware implementation of the CMA algorithm

It is possible to see that this implementation presents a reduction in the required hardware, when compared with the previous solutions, as the used basic blocks are simply multipliers. The delays $D1$ and $D2$ are introduced to ensure that the error function

is estimated considering the inputs and the respective outputs after the filtering, by compensating the delay introduced in the multipliers. With this step, the data modulation is removed and the error signal is constituted only by a phase error, the difference between the phase of the output signal and the input signal, and by an amplitude gain.

The hardware requirements, in terms of basic blocks, are presented in Table 6.7.

	Number	Primary blocks
		Multipliers
Complex multiplier	4	12
Complex – Real multiplier	2	4
$\|Complex\|^2$	1	2

Table 6.7 Basic blocks used in the implementation of the CMA algorithm

The total hardware requirements, after implementation, are presented in Table 6.8.

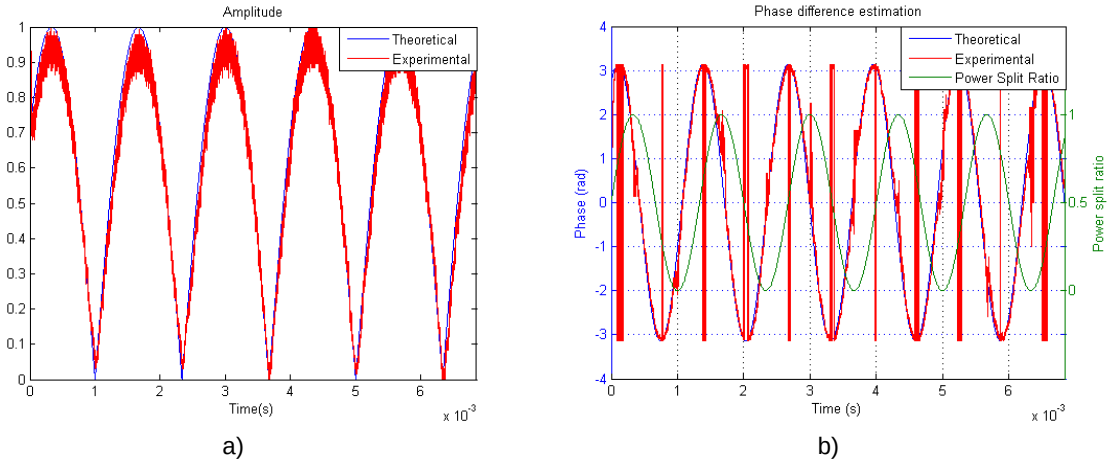
	Slice LUTs	Slice Registers	Slices	DSP48E1s
Total	323	172	100	18
Total (%)	0.21%	0.06%	0.27%	2.34%

Table 6.8 Total hardware requirements for the CMA implementation

If these results are compared with the hardware requirements of the Feedback algorithm, presented in Table 6.6, it is possible to see that the use of Slices has been reduced to half (reduction of 100 Slices). Also, this implementation requires 2 less DSP Slices, due to a reduction of 2 multipliers, and no LUT is used, while in the Feedback algorithm this was required to produce $e^{j\delta}$. However, the Feedback algorithm, with few additional resources, is able to track frequencies more than 50 higher than the CMA.

6.4.2. **Experimental results**

The estimated parameters, when considering slowly varying parameters (since the circuit is only able to track very slow variations), are presented in Figure 6.11.



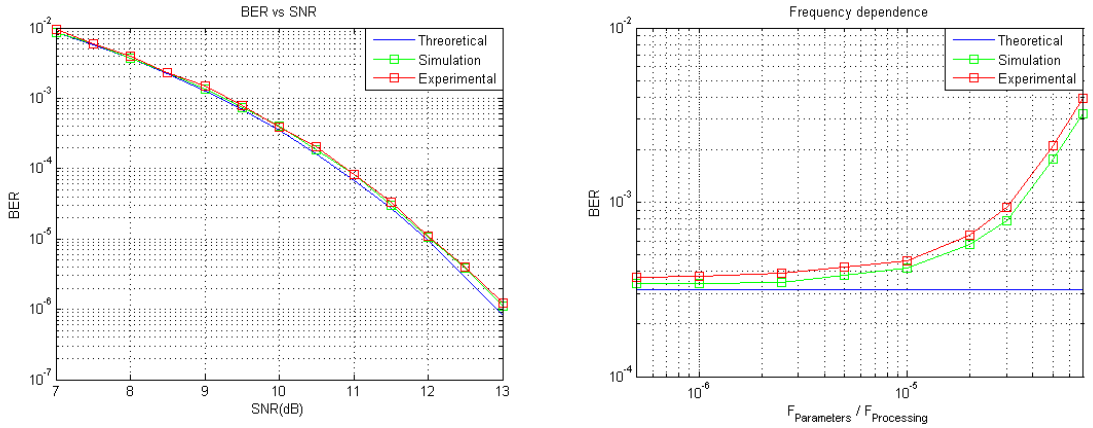
a) b)

Figure 6.11 Experimental results for a) amplitude $\sqrt{\rho}$ and b) phase difference δ using the CMA algorithm

It is possible to see that the obtained results follow closely the expected values. In Figure 6.11a) the results follow closely the expected values when ρ is close to zero, but when ρ is close to one the estimations present considerable noise, as expected from

simulation. The estimated phase difference δ, Figure 6.11b), only deviates from the expected values when one of the polarizations has no power, similar to the results obtained in simulation.

Again, a MC analysis was performed, considering frequencies of variation inferior to 800Hz to the parameters ρ and δ. The obtained results are presented in Figure 6.12a). Also, a frequency analysis was performed, and the obtained results are presented in Figure 6.12b).



a) b)

Figure 6.12 a) Experimental BER results for CMA algorithm considering frequency of variation of the SOP lower than 800Hz and b) frequency dependence

In Figure 6.12a). the experimental results follow very closely the expected values. However, we must keep in mind that this system was tested while the SOP presents a very low frequency of variation, and for this reason these results cannot be compared to the ones obtained for the other circuits, where variations of $50kHz$ are considered. From Figure 6.12b), the obtained results are close to the expected. However, let us not forget that the frequencies considered are by far inferior to that considered in the other algorithms.

6.5. Conclusions

As seen in the previous chapters, the CMA algorithm is the one that presents smaller hardware requirements. However, it is only able to track low frequencies, and should not be used in a real situation, where specifications impose that the selected circuit must be able to track variations of some kHz in the SOP of the received signal. Not considering the CMA algorithm, the Feedback implementation is the one that requires less hardware (even if compared with the CMA, a small increase in the required resources takes place) and, in terms of the estimated parameters, is the one that provides the most accurate results, as concluded by the MC analysis, when considering variations of $50kHz$ in the received SOP. For these reasons, the Feedback circuit is superior to the other options and is the right choice to implement the circuit considering the real case scenario, presented in the next chapter.

7. Real case scenario

7.1. Algorithms and simulation

In the real case scenario, as seen in the specifications, in each clock cycle 8 samples are received simultaneously for each polarization, corresponding to 4 received symbols sampled at 2 samples per symbol. Based on that information, the polarizations must be aligned, considering DOP=8. One could use only one of the samples to estimate the parameters ρ and δ in each clock cycle and disregard the others. However, some of the samples can correspond to inter-symbol transitions, which means that it is possible that the chosen sample always corresponds to a symbol transition, containing wrong information. To prevent this situation, and to increase the performance and robustness of the circuit against input noise and signal deformations, the estimations are based in averages, considering all the 8 received samples. This will lead to a high hardware replication, which may be problematic due to the finite available resources.

Remembering Figure 6.7, the signal E_x after the phase compensation, $E_x^1 = E_x e^{-j\delta}$, is extracted from the feedback loop implemented to estimate δ. If the amplitude compensation is also introduced in this feedback loop, without affecting its performance, the extracted signal, $\sqrt{\rho}E_x e^{-j\delta}$, will be completely compensated, both in amplitude and phase. However, remembering the previous chapters, the amplitude estimation goes through an inferior threshold and, if below that value, it is considered to be zero. If this signal was introduced in the feedback it would stop the loop. An alternative would be introducing the "raw" amplitude estimation in the loop and, at the output, correct the amplitude compensation with a simple conditional circuit. This move will avoid additional replication in the hardware, since the multipliers used in the phase compensation perform amplitude compensation as well.

After the necessary changes and replications, the obtained Feedback circuit is shown in Figure 7.1.

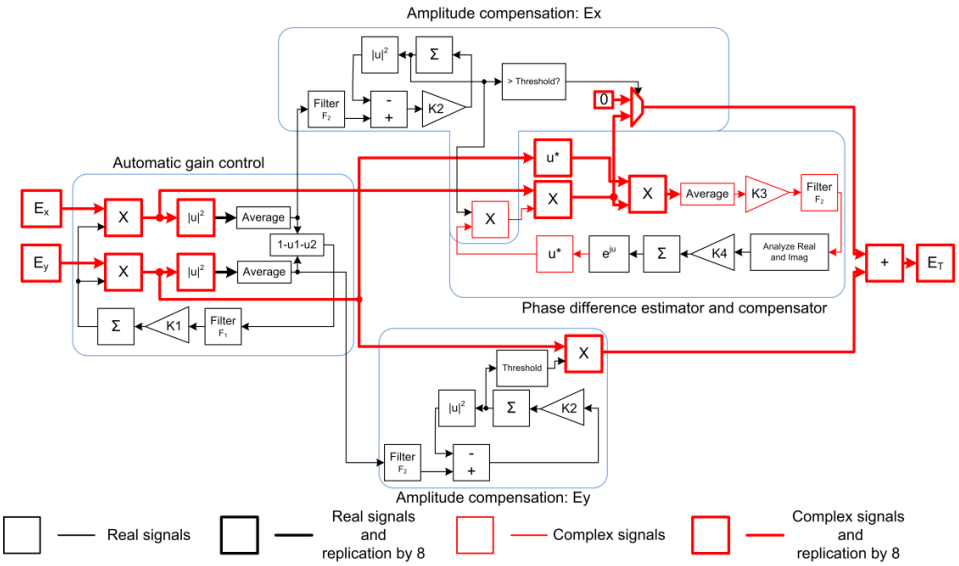


Figure 7.1 Final Feedback circuit, considering DOP=8

The circuit was tested using the data generated in section 4.2.2, considering that one of the two samples per symbol corresponds to the ideal sampling point, and the other falls in the middle of the inter-symbol transition. The estimations performed by the circuit are presented in Figure 7.2.

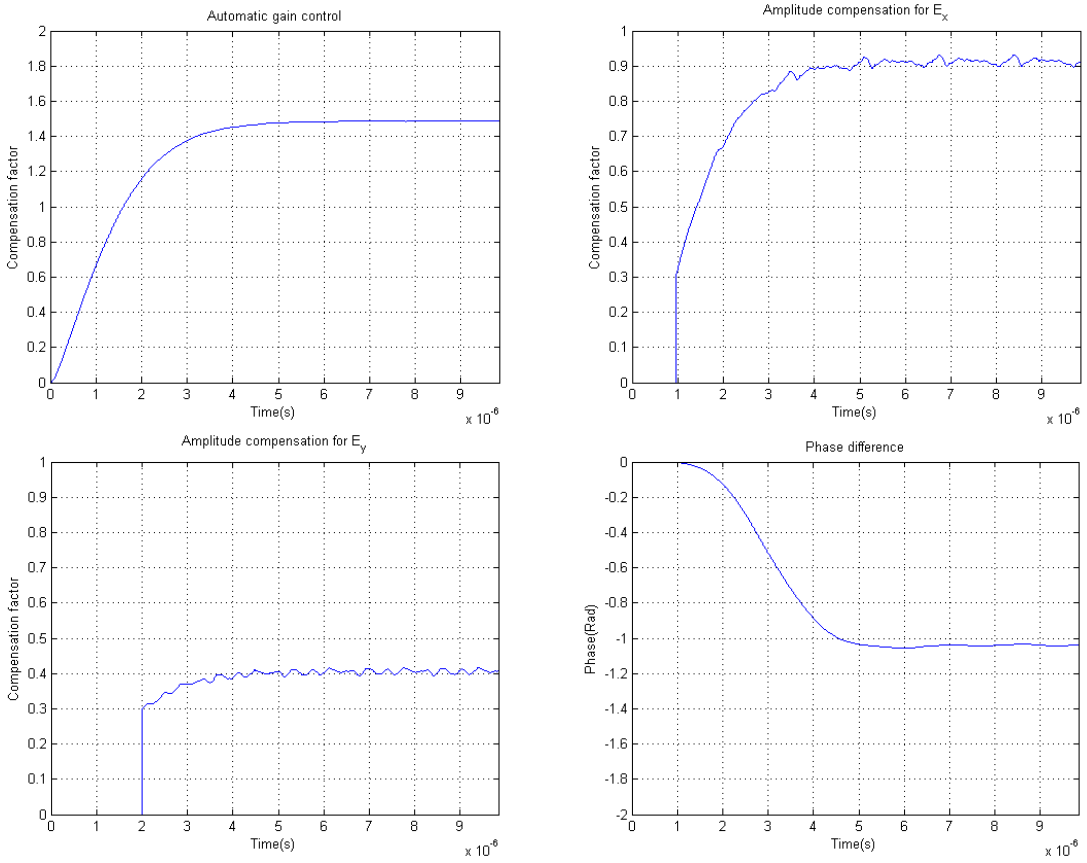


Figure 7.2 Parameters estimation using the circuit shown in Figure 7.1

It is possible to see that in less than $5\mu s$ the estimations are close to their final value. The phase estimation takes some time to converge to the correct value due to the required convergence of the automatic gain control algorithm and of the amplitude estimation for E_x, which start at zero. If these parameters were designed to start with the initial value of 1, the convergence would be much faster. The amplitudes estimations are zero until the value is higher than the inferior threshold considered. The parameters obtained after convergence were used to correct the input signals. Different sampling points were considered and the obtained results agree with the ones presented above. The estimated values for the different parameters, after the convergence time, are:

- $g_{agc} = 1.49$;
- $\sqrt{\rho} = 0.91$;
- $\sqrt{1-\rho} = 0.41$;
- $\delta = -1.04rad$.

If the correct sampling point is used and the parameters presented above are used to perform the equalization, it is possible to obtain the resulting constellation after compensation shown in Figure 7.3.

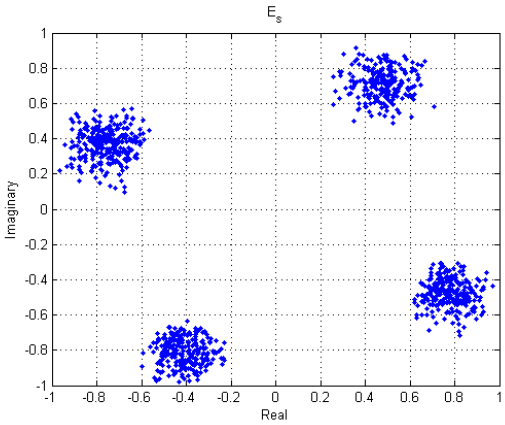


Figure 7.3 Constellation after compensation

The output signal has the phase of E_y, shown in Figure 4.12, which means that the signal E_x is correctly rotated. It is also possible to notice that amplitude equalization was performed.

7.2. Hardware implementation

As seen before, the implementation of the circuit requires a high hardware replication. The number of basic blocks used is indicated in Table 7.1.

	Number	Primary blocks	
		Multipliers	Sin & Cos LUT
Complex multiplier	16	48	0
Complex – Real multiplier	25	50	0
$\|Complex\|^2$	16	32	0
$\|Real\|^2$	2	2	0
Sin & Cos	1	0	1

Table 7.1 Basic blocks used in the implementation of the circuit shown in Figure 7.1

The total hardware requirements for the circuit are shown in Table 7.2.

	Slice LUTs	Slice Registers	Slices	DSP48E1s	RAMB36E1
Total	1199	616	353	132	1
Total (%)	0.8%	0.2%	0.94%	17.19%	6.25%* / 0.24%**

Table 7.2 Total hardware requirements for the circuit shown in Figure 7.1 (*when relative to the number of RAMB36E1 and ** if relative to the total available memory)

Again considering the data generated in section 4.2.2, the results obtained with the hardware implementation are shown in Figure 7.4.

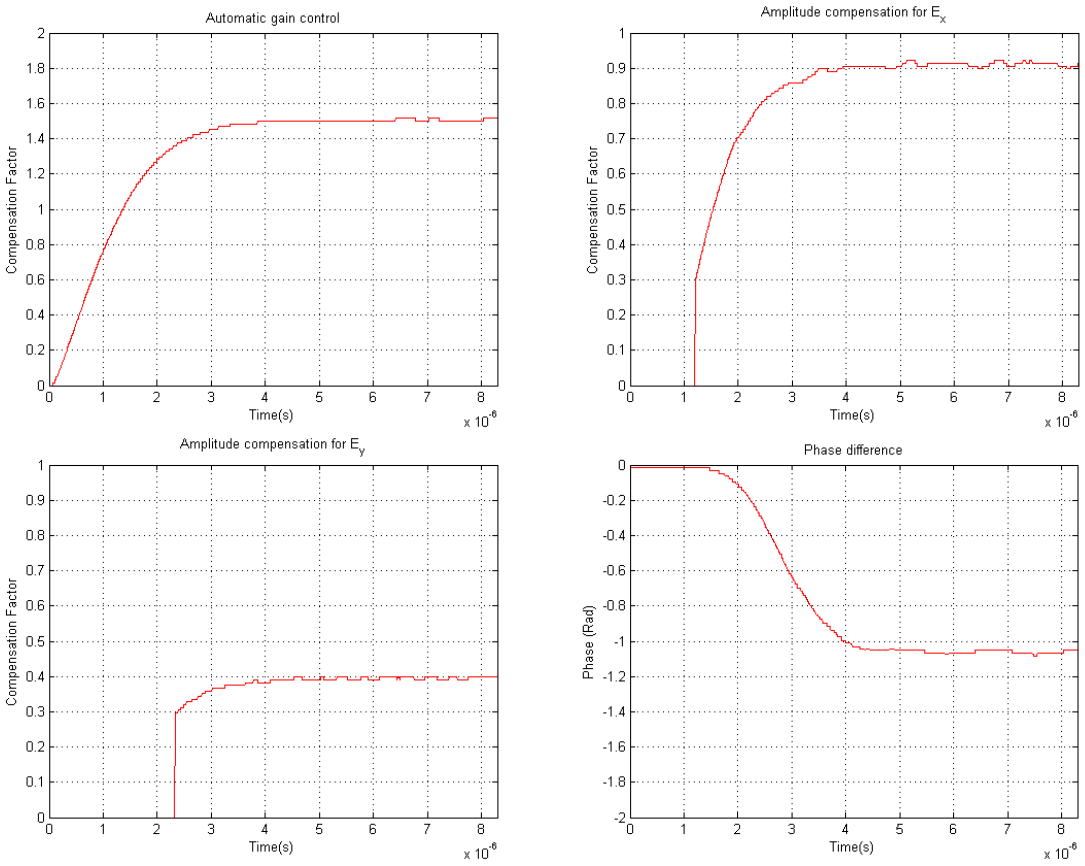


Figure 7.4 Parameters estimation using the hardware implementation of the circuit presented in Figure 7.1

The obtained results cope with the values obtained in simulation, demonstrating the correct behavior of the circuit.

7.3. Implementation considering five channels

As seen in Table 7.2, the Feedback implementation relies almost exclusively on the use of multipliers, implemented in the DSP slices. Less than 20% of the resources are taken for each case, which means that it is possible to implement 5 equivalent circuits processing in parallel, correcting 5 different channels. However, this solution requires many resources.

In practical situations the variation of ρ and δ is slower than the considered here, around some kHz, which means that the same circuit can be operated in sub sampling, compensating simultaneously the 5 channels. This solution was not implemented due to lack of time. However, a hardware study was performed to estimate the required resources. The multipliers used to compensate the input samples must be replicated, so that all the input samples are corrected. Hardware that performs operations over samples, such as filters and integrals, must also be replicated for each channel. The blocks or buses that require replication by 5 are represented using doted lines in Figure 7.5. Some parts of the circuit, such as the blocks used in the estimation of the parameters, can be shared by the 5 channels, resulting in considerable hardware savings. These blocks are represented using full lines in the same figure, and select one of the five channels to be processed in every clock cycle. One in every consecutive five samples is selected from

each channel to estimate the parameters, resulting in an equivalent processing frequency of $155.52/5MHz$, which is not problematic if assumed that the variation of ρ and δ is of some *kHz* (in the worst case, this circuit is now able to track variations of $10kHz$) in the worst case. However, all the samples must be compensated for all the channels simultaneously.

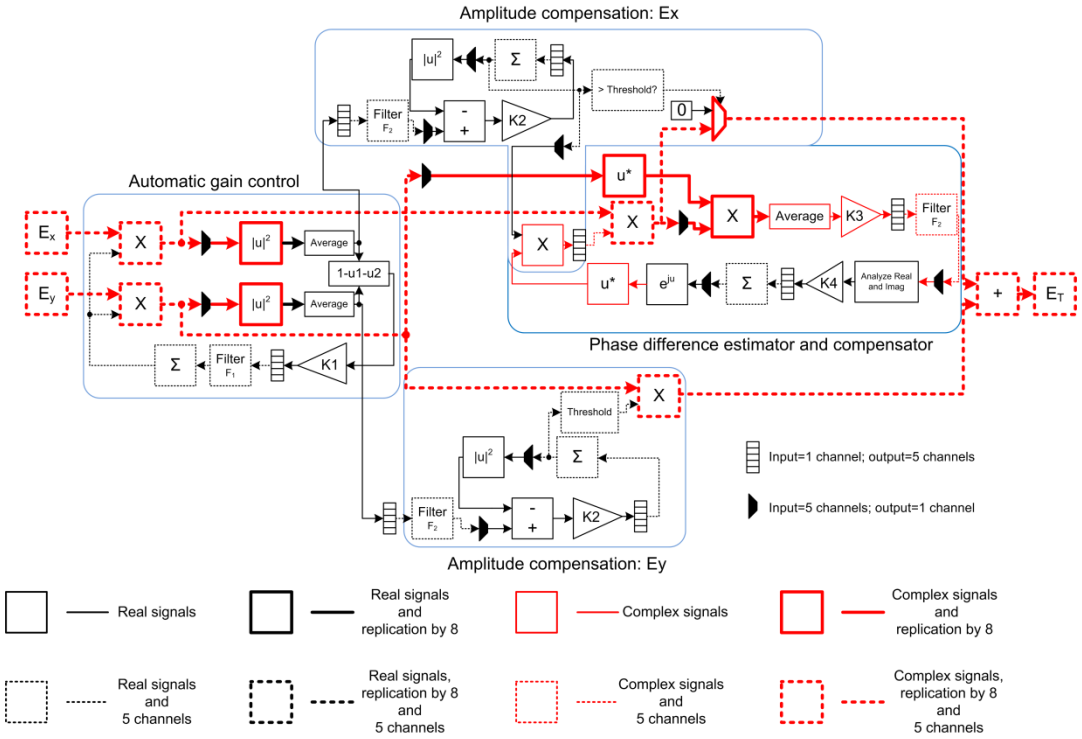


Figure 7.5 Final Feedback circuit prototype considering sub sampling for 5 channels

This circuit requires a complex signaling system, since the processed samples in the shared circuits must be forwarded to the correct filters and multipliers.

The expected number of basic blocks required to implement the circuit shown in Figure 7.5 is presented in Table 7.3.

	Number	Primary blocks	
		Multipliers	Sin & Cos LUT
Complex multiplier	16	48	0
Complex – Real multiplier	121	242	0
$\|Complex\|^2$	16	32	0
$\|Real\|^2$	2	2	0
Sin & Cos	1	0	1

Table 7.3 Basic blocks used in the implementation of the circuit shown in Figure 7.5

In this case a total of 324 multipliers is required, leaving considerable resources free to perform other operations. This circuit was not simulated nor implemented in FPGA due to lack of time. This task should be performed as future work. There is no concise information about the slice occupation of this circuit. However, a good idea of the required hardware can be obtained through the previous table.

8. Additional algorithms

While developing the polarization tracker algorithm, two other algorithms were developed and simulated, considering the same symbol-wise concept used in previous simulations. The working principle and some results for these algorithms are presented in the next subchapters.

8.1. SNR estimator

There are several ways of estimating the SNR of signal after the polarization combiner. To measure the SNR in a QPSK signal, the M2M4 algorithm [54] can be used. However this implementation requires more hardware than a solution that can be implemented using the developed circuits. In [55] a good technique is used to measure the OSNR. By rotating the polarization of the received signal, the system aligns the incoming polarization with the receiver and is able to measure the power of the signal plus noise. Then, by rotating the received polarization by 90º, it is able to detect only the received noise. With these two measures the system is able to measure the OSNR. Here the same concept can be used to measure the SNR of the signal after the polarization combiner which, if the circuit has an ideal performance, will be 3dB above the total SNR of the input signal. If the rotation matrix (5.13) is used, the first result of the rotation will be the signal plus noise (5.14) and the second output will measure only the noise,

$$N_T = -\sqrt{1-\rho}E_x + \sqrt{\rho}E_{y=} - \sqrt{1-\rho}N_x + \sqrt{\rho}N_y. \quad \textbf{(8.1)}$$

Since

$$E[N_T^2] = (1-\rho)E[N_x^2] + \rho E\left[N_y^2\right] = P_N \quad \textbf{(8.2)}$$

it is possible to obtain the noise power from this second output. By using the two outputs, (5.14) and (8.1), it is possible to obtain the SNR of the output signal,

$$SNR = \frac{E\left[E_T^2\right] - E[N_T^2]}{E[N_T^2]}. \quad \textbf{(8.3)}$$

This result can be obtained with the circuit presented in Figure 8.1.

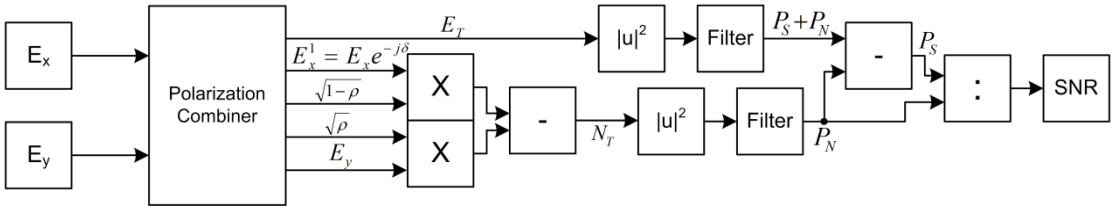


Figure 8.1 Block diagram for the SNR meter

Since the Feedback algorithm has proved to present the best results, this implementation was used for the polarization combiner. In this circuit the estimated phase

has a small delay when compared with the ideal value, which grows with the frequency of variation of δ. This means that some portion of the signal will be considered noise to the estimator. With this in mind, it is expected that the SNR is underestimated for higher variation frequencies of the estimated parameters. Also, the amplitude estimation meets some special criteria when the power is under a certain threshold, as seen before. In fact, some simulations shown that the use of the "raw" estimation for the amplitudes, without the inferior threshold limitation, resulted in better SNR estimations. This happens because, in (8.1), one of the products would become zero and the estimated noise would be increased using signal power.

Different situations are considered to study the influence of the variation frequency for the parameters and of the power distribution limits:

- Worst case: $0 < \rho < 1$, $-\pi < \delta < \pi$ and frequencies close to $50kHz$ for both parameters;
- Constraint ρ: $0.2 < \rho < 0.8$, $-\pi < \delta < \pi$ and frequencies close to $50kHz$ for both parameters;
- Low Freq: $0 < \rho < 1$, $-\pi < \delta < \pi$ and frequencies close to $15kHz$ for both parameters
- Best case: $0.2 < \rho < 0.8$, $-\pi < \delta < \pi$ and frequencies close to $15kHz$ for both parameters.

Different values of SNR were considered to the input signal, and results obtained under the different conditions are presented in Figure 8.2.

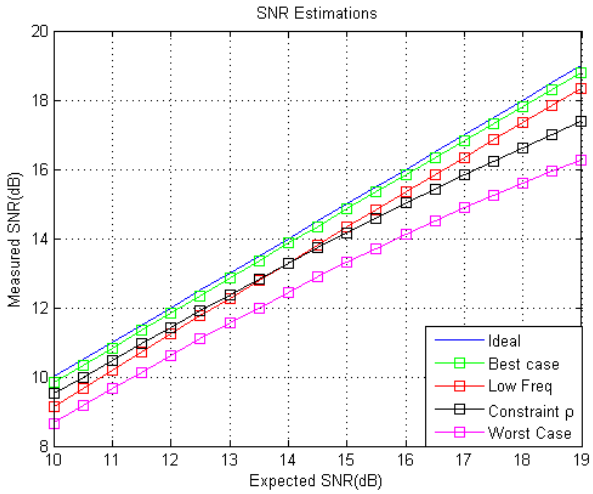


Figure 8.2: Simulation results for the SNR estimator considering different scenarios

As expected, there are several conditions where the estimated SNR falls far from the ideal values. The frequency of variation of both δ and ρ is the factor with the highest influence. A simple reduction, from $50kHz$ to $15kHz$, results in an accurate SNR estimation, with a penalty of less than 1dB facing the ideal result. The amplitude limitations also have a considerable impact in the results.

When high frequency of variation is considered for δ and ρ, the SNR penalty is bigger for higher SNRs of the input signal. This happens because the errors in the parameters estimations lead to an incorrect rotation, which will lead some of the signal power to be erroneously measured as noise. For high SNRs this factor has small impact,

since the estimated noise already has a considerable power. However, when small noise is present, this component may get higher than the noise itself, and degrades the SNR estimations.

8.1.1. **Hardware implementation**

Based in the block diagram of Figure 8.1, the SNR estimator was implemented in hardware. Additionally to the hardware required to implement the Feedback algorithm, the implementation of the SNR estimator required the blocks presented in Table 8.1.

	Number	Primary blocks	
		Multipliers	Dividers
Real division	1	0	1
Complex – Real multiplier	2	4	0
$\|Complex\|^2$	2	4	0

Table 8.1 Basic blocks used in the implementation of the SNR estimator

The hardware resources used by the SNR estimator are shown in Table 8.2.

	Slice LUTs	Slice Registers	Slices	DSP48E1s
Total	817	956	387	8
Total (%)	0.54%	0.32%	1.03%	1.04%

Table 8.2 Total hardware requirements for the implementation of the SNR estimator

The experimental results obtained are shown in Figure 8.3.

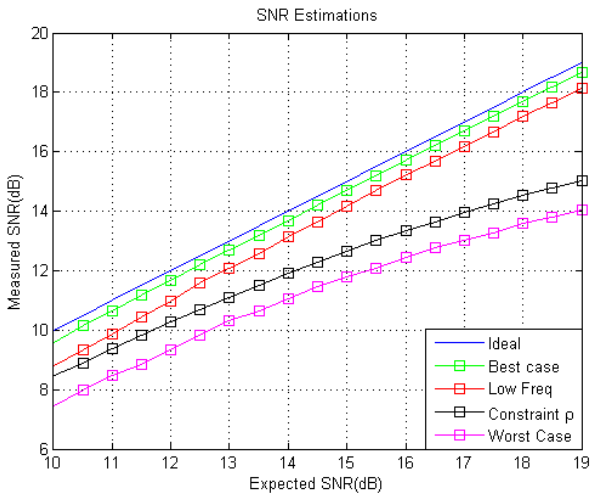


Figure 8.3 SNR estimations using the hardware implementation

For the cases where variations of $50kHz$ are considered, the obtained results are far from those obtained in simulation. This was an expected result, since delays introduced in the estimations, due to the hardware, increase the estimated noise signal, which decreases the estimated SNR. For the other cases, however, the results are very close to the ones obtained through simulation. In fact, for such conditions the error in the estimated SNR is small and constant, and could be corrected using a simple calibration.

8.2. Viterbi & Viterbi using Feedback

The different algorithms, used to perform the polarization tracker and combiner, were developed considering DQPSK modulation. If a non-differential code was used the phase error would have to be corrected before detection. There are different approaches to the Viterbi & Viterbi algorithm, in a completely feedforward implementation for the original circuit, or recurring to feedback [20] to be able to track higher and faster phase errors. Here, a different algorithm was developed, based in the same principles but using the concept that was applied in the Feedback algorithm to estimate δ.

To remove the phase modulation present in the signal, the fourth power is used in the original Viterbi & Viterbi algorithm [13]. The phase error can then be obtained through

$$\phi = \frac{1}{4}\arg((E_T)^4). \tag{8.4}$$

The same principle will be used here, now applied to a feedback loop. The proposed circuit, considering QPSK modulation, is shown in Figure 8.4.

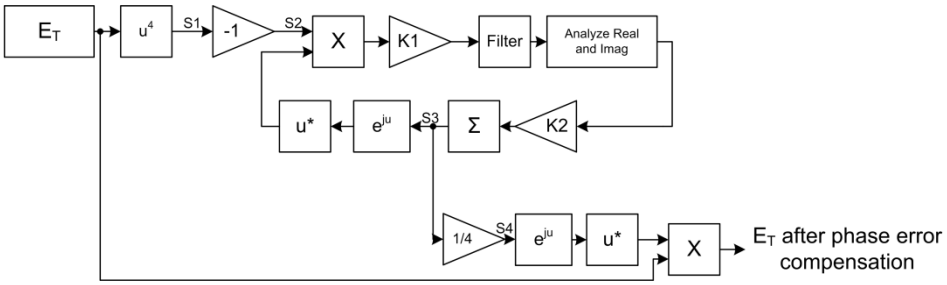


Figure 8.4 Block diagram for the phase error correction

Considering that the phase of the signal E_T is

$$\varphi = k\frac{\pi}{2} + \frac{\pi}{4} + \phi,\ k = 0,1,2,3 \quad , \tag{8.5}$$

where ϕ represents the phase error, the phase modulation is removed by the fourth power, in $S1$. However, since $\frac{\pi}{4}4 = \pi$ an additional gain of -1 must be introduced. In $S2$, a complex signal with the phase noise, amplified by four, is present. After convergence by the feedback loop, this amplified phase error will be present in $S3$.The signal $S4$ will be the resulting phase noise, and can be used to correct the input signal. In this case, the phase in $S3$ must be tracked between -8π and 8π so that, when dividing it by four, it can cover all the possible phase errors.

Although correcting the phase error introduced in the signal, a cycle sleep can be introduced, depending on the initial conditions. To correct this situation, a training sequence must be used at the beginning.

In order to have a fast response to phase variations, a high gain must be used in the feedback loop. However, if the gain is too high the input noise will interfere in the estimated phase, degrading the output signal. As a result of the small processing frequency considered, $155.52MHz$, the circuit is unable to correct high variations in the phase noise. Its performance is close to the one presented by the original Viterbi & Viterbi

algorithm, with the advantage of tracking phase errors above $\frac{\pi}{2}$, something that the original algorithm could not perform. To test this circuit, it was connected to the output of the polarization combiner (considering the Feedback algorithm). In this case, variations of $10kHz$ in the state of polarization of the received signal are considered. Higher frequencies could result in cycle sleeps in the output signal. The constellations at the different stages of the test are presented in Figure 8.5.

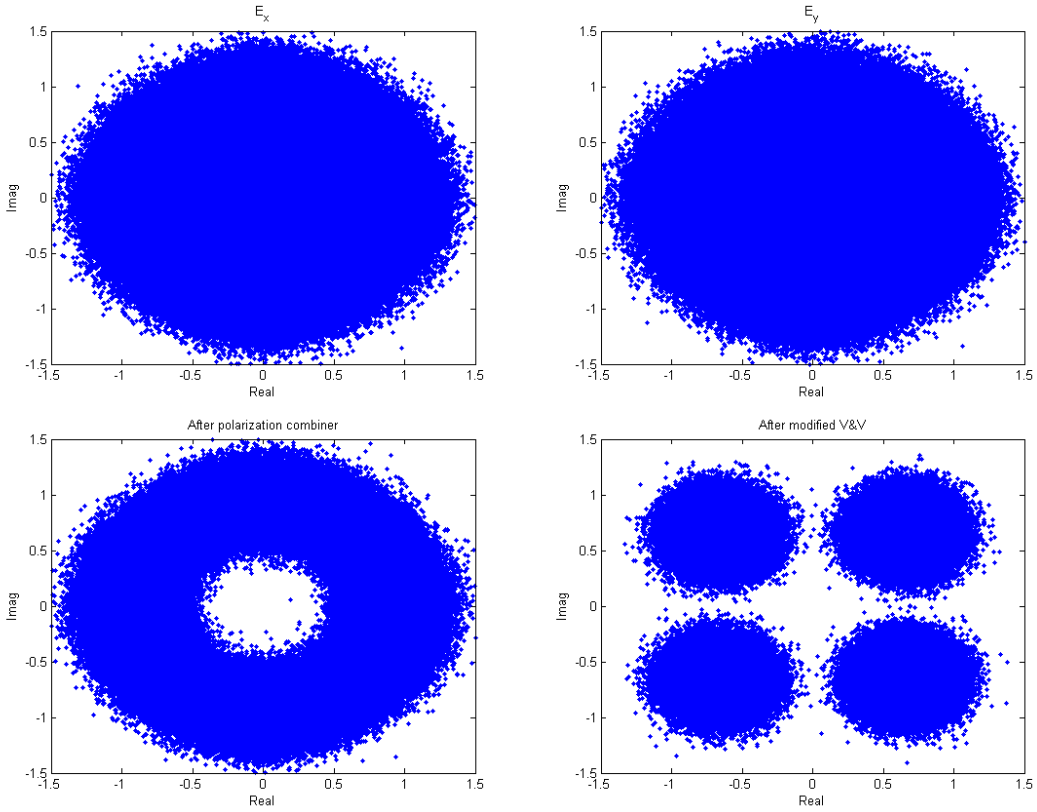


Figure 8.5 Constellations at the input, after polarization combiner and after phase error correction.

The input constellations, after a great number of samples, and due to the noise, result in a completely closed constellation due to the amplitude changes and phase rotations. After the polarization combiner, the resulting signal still suffers from phase rotations, but the amplitude is now constant. After this phase error is corrected, a good constellation is obtained.

Using the same concept, it is possible to obtain the performance of the circuit for different noise levels and use now non-differential detection for the received QPSK signal, comparing with the DQPSK limit. The results are shown in Figure 8.6.

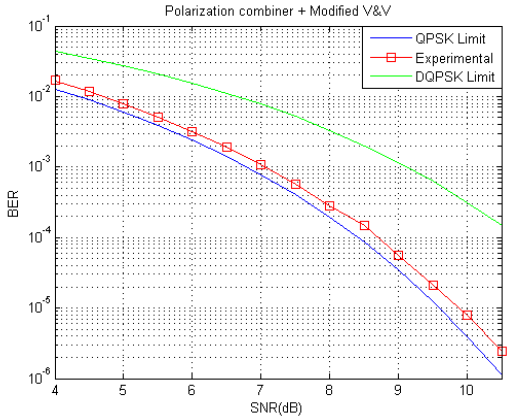


Figure 8.6 Simulated performance of polarization combiner and modified Viterbi & Viterbi algorithm

The use of QPSK modulation, combined with phase error correction, presents much better results than the use of DQPSK modulation. However, let us not forget that here no phase error was considered. In fact, after some tests, this circuit was observer to be able to track the phase error in systems where $LB/F_s = 10^{-7}$, where LB represents the bandwidth of the lasers used at the transmitter and at the receiver, and F_s the sampling rate. This performance, slightly superior to the obtained using the original Viterbi & Viterbi algorithm, is not enough to cope with real lasers, which have bandwidths between $100kHz$ and $10MHz$ [4]. This means that the previous algorithm is only useful to correct a slow varying phase error, and additional algorithms are required to correct phase noise. It is in this scenario that the use of differential coding, even with the associated SNR penalty, is useful, since the small phase jumps do not need to be corrected, as they do not greatly disturb the result.

8.2.1. **Hardware implementation**

The modified Viterbi & Viterbi algorithm was also implemented in hardware. The hardware requirements in terms of basic blocks are shown in Table 8.3.

	Number	Primary blocks	
		Multipliers	Sin & Cos LUT
Complex multiplier	4	12	0
Sin & Cos	0	0	2

Table 8.3 Basic blocks used in the implementation of the modified V&V algorithm

The total hardware requirements are presented in Table 8.4.

	Slice LUTs	Slice Registers	Slices	DSP48E1s
Total	491	81	175	12
Total (%)	0.33%	0.03%	0.46%	1.56%

Table 8.4 Total hardware requirements for the implementation of the SNR estimator

With the original Viterbi & Viterbi algorithm in mind, the hardware requirements for this implementation are slightly superior. However, there is no need to use the $arg()$ function, required by the implementation of the original algorithm.

Considering the same conditions as used in simulations, Figure 8.7 was obtained, where the results coincide with the ones obtained in simulation.

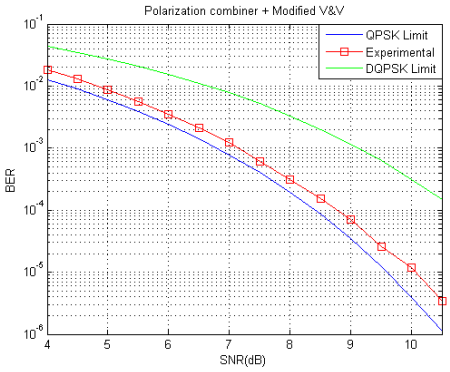


Figure 8.7 Experimental performance of polarization combiner and modified Viterbi & Viterbi algorithm

Still, let us not forget that this circuit by itself is not able to correct the phase noise present in optical systems, and for this reason DQPSK modulation is the best option when low sampling frequencies are used.

9. Conclusions

In this document, digital post compensation for PMD when single polarization is used was studied. Although not a very complex problem as it would be if PDM was used, where polarization scrambling would occur, high frequency variations in the SOP of the received optical signal can compromise the performance of the receiver, depending on the algorithm used to perform polarization tracking and combining. Different approaches to solve this problem were simulated and implemented in hardware. The best circuit was selected, considering both performance and hardware requirements.

In chapter 2 a small introduction to optical coherent systems and fiber propagation impairments was performed. It was seen that the effects of PMD, in the adopted scenarios, can be simulated at the receiver input by considering the electric field as the sum of two orthogonal polarization components that are combined by a simple power splitting and phase rotation between two copies of the modulated signal. This principle, confirmed in chapter 4 through the simulation of the specified scenario in OSIP, was used to create a simplified channel simulator in SIMULINK, in which the introduced impairments were simulated using sinusoids to test the algorithms capability to estimate and compensate parameters varying over time. A small introduction to the development environments used for the simulations and hardware tests was performed. A simple RS232 interface was developed to implement the communications between MATLAB, where the data was simulated and analyzed afterwards, and the ML605 evaluation kit, where the information was processed.

In chapter 5 different approaches were studied to the implementation of a PMD equalizer. Starting with the CMA algorithm adapted to single polarization signals, unable to cope with the specifications, a Feedforward solution was tested. This algorithm presented severe limitations and a new Feedforward implementation was developed and tested. However, even this second algorithm, in the two implemented versions, presented hardware disadvantages, as seen in chapter 6. For this reason, and considering the required hardware, a Feedback algorithm was developed.

In chapter 6 the developed algorithms were implemented in hardware. Again, the Feedback algorithm presented the best performance. In addition, in terms of hardware requirements, the Feedback approach turned out to be the best option, when compared with the Feedforward algorithms. Even when compared with the CMA algorithm, the Feedback requires few additional resources and is able to track higher frequencies of variation for the received SOP. Through the use of feedback loops some nonlinear and high hardware-intensive operations can be performed through some multipliers and LUTs. Being the best option, the Feedback solution was selected to implement the algorithm to meet the real case specifications, which had been introduced in chapter 3 when the NGOA system was studied.

In chapter 7, with the required degree of parallelization of 8 in mind, a solution was found to perform the possible simplifications in the hardware replication. The algorithm was analyzed using the samples retrieved from OSIP, to test its performance in conditions close to what it would face in a real implementation. The estimations for the parameters converged to the final values in under $5\mu s$, and the signal was correctly equalized.

In chapter 8 two other algorithms were introduced. Through some additional blocks it was possible estimate the SNR of the output signal. The results showed that, if the channel impairments have a small variation frequency, the estimated SNR follows closely the expected values. Also, combining the feedback loop used to estimate the phase difference, in the Feedback algorithm, with the Viterbi & Viterbi algorithm, it is possible to obtain a modified Viterbi & Viterbi algorithm. This modified circuit presents a better performance than the original Viterbi & Viterbi implementation with a small resource increase, but still is unable to correct phase noise introduced by the lasers, showing the need for differential coding when such small sampling rates are used.

9.1. Future work

There are some activities that could follow this work in the near future.

The implementation of a single circuit to equalize five different channels was simply studied in terms of hardware requirements. It would be interesting to develop and implement the actual circuit, since this approach translates into high hardware reduction when compared to the replication of the total hardware by the number of channels.

Additionally, the circuits were tested using data retrieved from simulations. The next step would consist in implementing and testing this solution using real data, in offline or in real time signal processing.

Finally, there are other algorithms that can be implemented in the same setup to complement the polarization combiner, such as the digital PLL for the local electrical oscillator or the DSP used to correct the sampling point.

10. Annex 1

In this annex a brief explanation is performed on how to obtain the basic blocks that constitute the different circuits, most of them requiring the use of IPCores.

Multiplier 16x16

The basic multipliers were obtained using the “Multiplier Generator 11.2” [56]. Since the inputs have 16 bits, the output of each simple multiplication will have 32 bits. From these 32 bits, the [26 11] (from the [31 0]) must be selected to the output so it has the desired Q4.11 format. The multipliers make use of the embedded multipliers of the FPGA, the DSP48E1 slices, and are optimized for speed, presenting a variable delay (pipeline stages). In this case two pipeline stages were used, although the optimal pipeline was of 3 delays. The DSP48E1 slices support multipliers up to 25 bits x 18 bits, which means that the desired representation leads to multipliers that can be mapped using a single DSP slice.

Using two of these basic multipliers and one adder it is possible to obtain the power of a complex number since

$$|(a+jb)|^2 = (aa+bb). \tag{10.1}$$

With the same hardware it is also possible to obtain a complex to real multiplier,

$$k(a+jb) = ka+jkb. \tag{10.2}$$

With four basic multipliers and two simple adders it is possible to obtain a complex multiplier since

$$(a+jb)(c+jd) = (ac-bd)+j(bc+ad). \tag{10.3}$$

However, it is possible to obtain a complex by complex multiplier using less resources through the “Complex multiplier v5.0” [57] since

$$(a+jb)(c+jd) = \big(a(c+d)-d(b+a)\big)+j(a(c+d)+c(b-a)), \tag{10.4}$$

meaning that the same operation can be implemented using only 3 multipliers and additional logic. Note that this logic can be problematic and lead to higher pipeline if high clock frequencies are used. However, the system proved to operate correctly without additional delays at the desired clock frequency.

The additional logic was implemented without delays, and the total multiplication process presents a delay of 2 clock cycles.

Complex conjugate

The complex conjugate of a complex number is obtained by calculating the symmetric of the imaginary component. Since the numbers are represented using the two's complement, it is possible to obtain the complex conjugate simply by negating the imaginary component. This task presents no cycle delay from the input to the output.

Divider

In order to have the desired 11 fractional bits at the output of a division, the dividend must have more 11 fractional bits than the divisor. Having this in mind, the dividend must be expanded by 11 bits (simply copying the signal bit to the 11 additional bits) and the division can be performed 27 bits / 16 bits. The [15 0] bits of the output will contain the result. The divider was obtained using the "Divider Generator 3.0" [58]. From the two available algorithms, the Radix2 algorithm, that calculates 1 bit of the result per iteration, was selected. The division presents a delay of 33 clock cycles at the output. To prevent the divisor from being zero, a conditional case can be applied: if the divisor is detected to be zero, it will be replaced by a small number, to prevent major errors in the output. However this does not correct the fact that, for small divisors, the result of the division may be incorrect due to overflows and due to the influence of the noise in these cases.

A simple divider has high hardware requirements and the synthesis tool was having problems synthesizing the VHDL code with the selected divider configured to run at the desired processing frequencies. A simple solution to the problem was found by reducing the sampling ratio of the divider by 2 (using 2 clocks per division cycle) and employing two dividers processing in parallel. When one of them is sampling the current input, the other is processing the last input. At the end of 33 cycles the correct output must be selected from the two dividers. With this solution, in Figure 10.1, the resulting hardware is almost doubled but the timing requirements are relaxed and the compiling tool was able to generate the configuration file without any troubles.

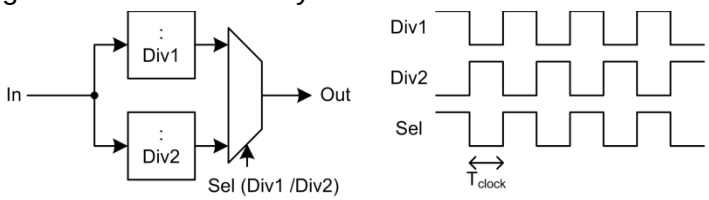


Figure 10.1 Divider implemented using two parallel dividers

When the dividend is a complex number, all the mentioned hardware will double since the division operations are performed independently for each component (resulting in a total of 4 dividers due to the previous parallelism).

Square root

The square root was implemented using the "CORDIC 4.0" generator [59]. A CORDIC (Coordinate Rotation Digital Computer) is an algorithm initially developed by Volder [60], and has the capability of solving iteratively operations, such as trigonometric equations or, in this case, a square root. A parallel implementation was used so that all the inputs can be processed, at the cost of silicon area. This block introduces a delay of 9 clock cycles.

The available CORDIC only supports unsigned inputs smaller or equal to 2. To solve this limitation, a binary shift (by a pair number of bits) is performed at the input, to ensure that the input value never rises above 2. At the output, the inverse operation is performed, but now the shift is done with half of the bits used before due to the square root. For example, if a right shift of 4 bits is done at the input, at the output a left shift of $\frac{4}{2} = 2$ bits must be performed.

IIR Filter

The used IIR filters can be implemented simply using a delay, an addition and two constant gains that can be "hardwired" in the VHDL code, not requiring multipliers. Since several different IIR filters will be used through this work, the generic filter in Figure 10.2 was designed.

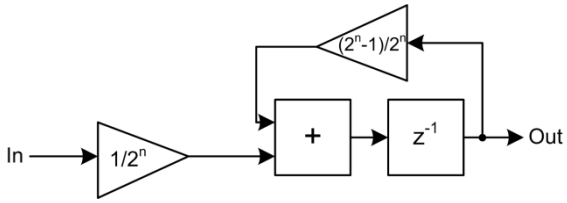


Figure 10.2 Block diagram of a generic filter

By changing the value of n it is possible to obtain a IIR filter with different coefficients. When filtering complex signals two filters are required, one for the real and one for the imaginary component.

Arg(u): Extract phase of complex signal using a CORDIC

To estimate the phase of a complex number the "CORDIC 5.0" generator [61], operating with the "Arc Tan" option, was used. It performs rotations until the imaginary component is zero, which means that the normal $\frac{\pi}{4}$ limitations of the arctangent function do not apply and the input signal can have an arbitrary phase (the "coarse" option was selected, allowing input values in any position of the unitary circle). The input does not have to be normalized which simplifies the circuit, and is one of the advantages of using this block. The output has 14 bits (Q3.11), enough to represent the phase information in the interval $]-\pi\ \pi]$.

The block introduces a delay of 16 clock cycles which will delay the estimated phase difference and may compromise the performance.

e^{ju}: Obtain complex signal from phase using a CORDIC

A complex exponential can be obtained again using a "CORDIC 5.0", now in the "Sin & Cos" option, since $e^{ju} = \cos(u) + jsin(u)$. The input phase has 14 bits (in the format Q2.11) and is expressed in radians, taking values from $-\pi$ to π, so the "coarse" option was selected. The output values have 13 bits (in the format Q1.11, saving all the information), and the signal bit must be propagated to the other 3 bits of the output to keep the format Q4.11. It must be made sure that the input phase is between $-\pi$ and π, or the results will be unknown. This can be performed with conditional statements: if the phase, θi, is bigger than π, the used input value will be $\theta i - 2\pi$ (bigger than $-\pi$), and, if it

is smaller than $-\pi$, the used input will be $\theta i + 2\pi$ (smaller than π). Since rotations of 2π radians are performed, the result won't be affected.

This block introduces a delay of 16 clock cycles.

e^{ju}: Obtain complex signal from phase using a LUT

The CORDIC used to perform the "Sin & Cos" operation, implemented through iterative logic, consumes a considerable amount of resources and introduces a considerable delay in the signal. A good alternative would be using a LUT to store some of the values. A LUT is a memory where the information present in a certain position is the result of the desired function when the input is the address used to access that memory slot.

In the present case, the desired functions are sine and cosine. To save resources, only one quadrant of the cosine function needs to be stored in memory and, using trigonometric identities, it is possible to obtain all the other cases. Table 10.1 shows how the cosine and sine of any angle can be obtained by finding the equivalent angle in the first quadrant (with the help of a signal change as well).

σ	$\left]-\pi \quad -\frac{\pi}{2}\right]$	$\left]-\frac{\pi}{2} \quad 0\right[$	$\left[0 \quad \frac{\pi}{2}\right]$	$\left]\frac{\pi}{2} \quad \pi\right]$
cos ()	$-cos(\pi + \sigma)$	$cos(-\sigma)$	$cos(\sigma)$	$-cos(\pi - \sigma)$
sin ()	$-cos\left(-\frac{\pi}{2} - \sigma\right)$	$-cos(\frac{\pi}{2} + \sigma)$	$cos\left(\frac{\pi}{2} - \sigma\right)$	$cos\left(\sigma - \frac{\pi}{2}\right)$

Table 10.1 Obtaining the cosine and sine of an angle by reducing it to the first quadrant

In hardware, the value $\frac{\pi}{2}$ was normalized to $1 - 2^{-11}$. This way, simply by using the fractional bits it is possible to access all the positions in the LUT from 0 to $\frac{\pi}{2}$. In the LUT, values from 0 up to 1 must be represented, since that is the excursion of the cosine in the considered quadrant. This means that the values stored in the memory have 12 bits, 11 fractional and 1 integer bit. No signal is necessary since the cosine only takes positive values in the first quadrant. Using a dual port ROM it is possible to perform two simultaneous accesses to the memory, which allows the estimation of sine and cosine in one clock cycle, whereas using a CORDIC 16 clock cycles were required. This will increase the performance of the circuit. The dual port ROM was obtained using the "Memory generator 6.2" [62] and was configured to have two outputs sharing the same clock and store 2048 12-bit words. The ROM will be mapped into a 36Kb block RAM available in the FPGA, called RAMB36E1. It is possible to estimate the number of slices used if the memory was implemented using slice LUTs. This memory requires

$$2^{11} * 12 = 24576\ bits. \quad \textbf{(10.5)}$$

In Virtex6, each Slice has 4 6-bit LUTs, meaning that each slice can be used to store

$$2^{6} * 4 = 256\ bits. \quad \textbf{(10.6)}$$

The total memory would then, in addition with the external logic, require approximately 100 Slices to be implemented. This represents half the number of slices required to implement the CORDIC, presented in Table 10.2 under the name “Sin & Cos”.

Hardware requirements for the basic blocks

The hardware requirements for the different basic blocks are presented in Table 10.2.

	Slice LUTS	Slice Registers	Slices	DSP48E1s	RAMB36E1
Simple Multiplier	1	0	1	1	0
Complex Multiplier	0	1	1	3	0
Square root	218	163	103	0	0
Divider	1296	1811	723	0	0
Atan	733	705	237	0	0
Sin & Cos	668	646	243	0	0
Sin & Cos LUT	0	0	0	0	1

Table 10.2 Hardware requirements of the different basic blocks

The dividers are the blocks that consume the higher resources, due to the duplication required to simplify the synthesis tool operations. Even if this step was not performed, more than 400 Slices were required to perform a simple division. The use of CORDICs to obtain trigonometric functions present similar requirements, close to 240 Slices. The multipliers practically only require the DSP slices.

11. Annex 2

Despite the presented algorithms, the first explored Feedforward solution was, in fact, the circuit presented in Figure 11.1.

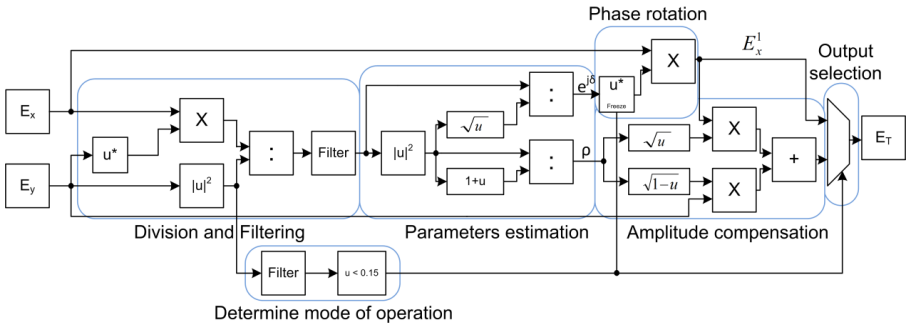


Figure 11.1 First attempt at the Feedforward algorithm

In this circuit, instead of tracking the phase difference δ as an angle, it is estimated using the real and imaginary components of $e^{j\delta}$, through the normalization

$$e^{j\delta} = \frac{\vec{r}(i)}{|\vec{r}(i)|}. \tag{11.1}$$

The same idea as used in the circuit presented in Figure 5.11 is also applied here, monitoring the power of E_y to select the correct output and freeze the last correctly estimated phase, before it is corrupted by the errors from the division. In this case, however, instantaneous phase jumps can occur, depending on the changes that δ suffers when the circuit is in the alternative mode of operation. If the difference between δ_{old} (the stored last correctly estimated phase difference) and the new δ is high enough, fast phase rotations occur and errors will arise at the demodulator. The estimated phase difference is presented in Figure 11.2.

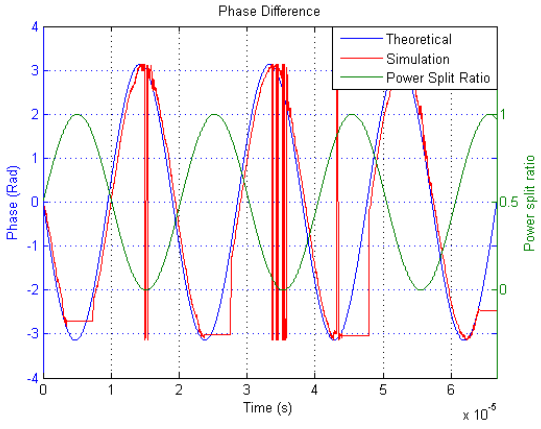
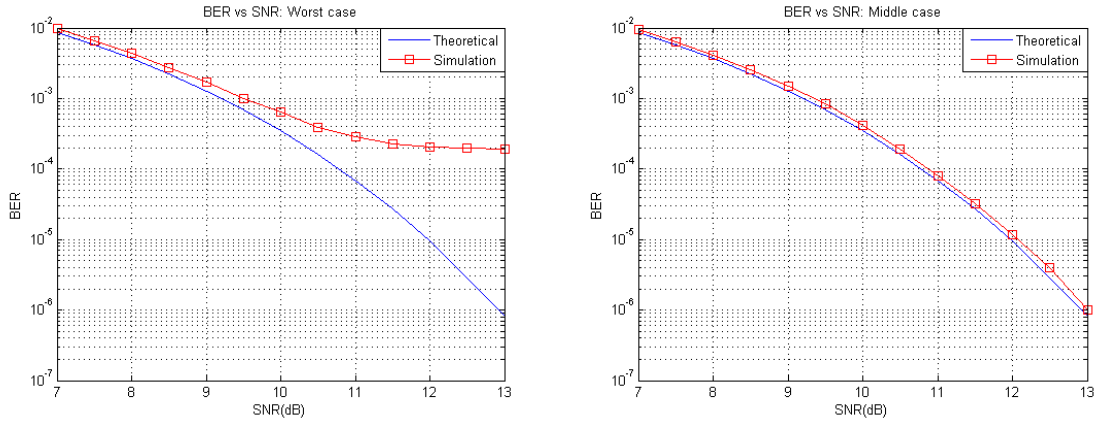


Figure 11.2 Simulation results for δ using the circuit shown in Figure 11.1

Note that now the high phase transitions presented by the estimated δ correspond to instantaneous phase jumps, while in Figure 5.12 these transitions occurred across several samples, due to the filter applied to δ. In this case no filter can be applied since,

while tracking a complex value, if a π phase jump occurred the filtered result would go across the unitary circle, possibly taking amplitudes close to zero.

In order to test the quality of the output signal, MC analyses were performed, one considering the worst case scenario, and a second one considering a middle case scenario, where ρ is constrained to $0.2 < \rho < 0.8$, to test the circuit in a situation where the power is never completely in one polarization, meaning that no instant phase jumps occur since δ is always correctly tracked. The results are present in Figure 11.3.



a) b)

Figure 11.3 Simulation BER results for a) the worst and b) a middle case scenario

In the worst case scenario, for low SNR values, the circuit's response is close to the limit. However, for higher SNRs the circuit presents an error floor at $BER = 2 * 10^{-4}$ due to the instantaneous phase jumps discussed above. In fact, in terms of symbols, the floor is present for $SER = 2.82 * 10^{-4}$, which is a little lower than the normalized frequency considered for ρ, $3.3 * 10^{-4}$. This means that not every context switch causes a symbol error, depending on the evolution of δ during the time the circuit is in the alternative mode of operation. For the middle case scenario, the results are close to the optimal values, showing that the circuit is working properly.

This circuit was also implemented in hardware, and its requirements are shown in Table 11.1.

	Slice LUTs	Slice Registers	Slices	DSP48E1s
Total	7503	9693	3178	14
Total (%)	4.98%	3.22%	8.43%	1.82%

Table 11.1 Total hardware usage for the first Feedforward approach

Additionally to the instantaneous phase jumps, which degrade the output signal, this implementation requires more resources, due to the additional division used to normalize $e^{j\delta}$ (in fact two divisions, since it is a complex signal), showing that this was not a good solution.

12. References

[1] G. P. Agrawal, *Fiber-Optic Communication Systems*: Wiley-Interscience, 2002.

[2] A. H. Gnauck, R. W. Tkach, A. R. Chraplyvy, and T. Li, "High-Capacity Optical Transmission Systems," *Lightwave Technology, Journal of,* vol. 26, pp. 1032-1045, 2008.

[3] A. H. Gnauck, G. Charlet, P. Tran, P. J. Winzer, C. R. Doerr, J. C. Centanni, E. C. Burrows, T. Kawanishi, T. Sakamoto, and K. Higuma, "25.6-Tb/s WDM Transmission of Polarization-Multiplexed RZ-DQPSK Signals," *Lightwave Technology, Journal of,* vol. 26, pp. 79-84, 2008.

[4] K. Kikuchi, *Digital Coherent Optical Communication Systems: Fundamentals and Future Prospects*: IEICE, 2011.

[5] H. S. C. Sun Hyok Chang, Kwangjoon Kim, "Compensation of front-end IQ-mismatch in coherent optical receiver," *Optical Internet Research Department, Electronics and Telecommunications Research Institute,,* 2011.

[6] K. Grobe and J. P. Elbers, "PON in adolescence: from TDMA to WDM-PON," *Communications Magazine, IEEE,* vol. 46, pp. 26-34, 2008.

[7] N. C. Leonid G. Kazovsky, Wei-Tao Shaw, David Gutierrez, Shing-Wa Wong, *Broadband Optical Access Networks*: Wiley-Interscience, 2011.

[8] F. Heismann, D. A. Fishman, and D. L. Wilson, "Automatic compensation of first order polarization mode dispersion in a 10 Gb/s transmission system," in *Optical Communication, 1998. 24th European Conference on*, 1998, pp. 529-530 vol.1.

[9] E. Ip, A. P. T. Lau, D. J. F. Barros, and J. M. Kahn, "Coherent detection in optical fiber systems," *Opt. Express,* vol. 16, pp. 753-791, 2008.

[10] T. Saito, N. Henmi, S. Fujita, M. Yamaguchi, and M. Shikada, "Prechirp technique for dispersion compensation for a high-speed long-span transmission," *Photonics Technology Letters, IEEE,* vol. 3, pp. 74-76, 1991.

[11] J. H. Winters and R. D. Gitlin, "Electrical signal processing techniques in long-haul, fiber-optic systems," in *Communications, 1990. ICC '90, Including Supercomm Technical Sessions. SUPERCOMM/ICC '90. Conference Record., IEEE International Conference on*, 1990, pp. 397-403 vol.2.

[12] M. G. Taylor, "Coherent detection method using DSP for demodulation of signal and subsequent equalization of propagation impairments," *Photonics Technology Letters, IEEE,* vol. 16, pp. 674-676, 2004.

[13] A. Viterbi, "Nonlinear estimation of PSK-modulated carrier phase with application to burst digital transmission," *Information Theory, IEEE Transactions on,* vol. 29, pp. 543-551, 1983.

[14] S. J. Savory, "Digital filters for coherent optical receivers," *Opt. Express,* vol. 16, pp. 804-817, 2008.

[15] H. Keang-Po and J. M. Kahn, "Electronic compensation technique to mitigate nonlinear phase noise," *Lightwave Technology, Journal of,* vol. 22, pp. 779-783, 2004.

[16] G. Charlet, "Progress in optical modulation formats for high-bit rate WDM transmissions," *Selected Topics in Quantum Electronics, IEEE Journal of,* vol. 12, pp. 469-483, 2006.

[17] T. Yoshida, T. Sugihara, H. Goto, T. Tokura, K. Ishida, and T. Mizuochi, "Fast feed-forward polarization tracking in a DP-mPSK coherent receiver employing blind SOP estimation," in *Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2012 and the National Fiber Optic Engineers Conference*, 2012, pp. 1-3.

[18] N. Mantzoukis, A. Vgenis, C. S. Petrou, I. Roudas, T. Kamalakis, and L. Raptis, "Design guidelines for electronic PMD equalizers used in coherent PDM QPSK systems," in *Optical Communication (ECOC), 2010 36th European Conference and Exhibition on*, 2010, pp. 1-3.
[19] K. Kikuchi, "Optical Homodyne Receiver Comprising Phase and Polarization Diversities with Digital Signal Processing," in *IEEE/LEOS Summer Topical Meetings, 2007 Digest of the*, 2007, pp. 55-56.
[20] M. Seimetz, *High-Order Modulation for Optical FiberTransmission*: Springer, 2009.
[21] G. P. Agrawal, *Lightwave Technology: Telecommunication Systems*: Wiley-Interscience, 2005.
[22] E. B. Sergio Benedetto, *Principles of Digital Transmission: With Wireless Applications*: Springer, 1999.
[23] L. E. Miller and J. S. Lee, "BER expressions for differentially detected π/4 DQPSK modulation," *Communications, IEEE Transactions on,* vol. 46, pp. 71-81, 1998.
[24] *Introduction to DWDM Technology*: Cisco Systems, 2001.
[25] C. R. M. Andrea Galtarossa, *Polarization Mode Dispersion*: Springer, 2005.
[26] J. L. P. Paulo S. André, "Birrefringência e Dispersão Devido aos Modos de Polarização em Fibras Ópticas," *Revista do Departamento de Electrónica e de Telecomunicações,* vol. 3, 2002.
[27] P. J. Winzer and R. J. Essiambre, "Advanced Optical Modulation Formats," *Proceedings of the IEEE,* vol. 94, pp. 952-985, 2006.
[28] P. Ciprut, B. Gisin, N. Gisin, R. Passy, P. Von Der Weld, F. Prieto, and C. W. Zimmer, "Second-order polarization mode dispersion: impact on analog and digital transmissions," *Lightwave Technology, Journal of,* vol. 16, pp. 757-771, 1998.
[29] A. O. Dal Forno, A. Paradisi, R. Passy, and J. P. von der Weid, "Experimental and theoretical modeling of polarization-mode dispersion in single-mode fibers," *Photonics Technology Letters, IEEE,* vol. 12, pp. 296-298, 2000.
[30] H. Bulow, W. Baumert, H. Schmuck, F. Mohr, T. Schulz, F. Kuppers, and W. Weiershausen, "Measurement of the maximum speed of PMD fluctuation in installed field fiber," in *Optical Fiber Communication Conference, 1999, and the International Conference on Integrated Optics and Optical Fiber Communication. OFC/IOOC '99. Technical Digest*, 1999, pp. 83-85 vol.2.
[31] P. M. Krummirich and K. Kotten, "Extremely fast (microsecond timescale) polarization changes in high speed long haul WDM transmission systems," in *Optical Fiber Communication Conference, 2004. OFC 2004*, 2004, p. 3 pp. vol.2.
[32] G. Agrawal, *Nonlinear Fiber Optics*: Academic Press, 2001.
[33] J.-P. Laude, *DWDM Fundamentals, Components and Applications*: Artech House Print on Demand, 2002.
[34] H. P. Yuen and V. W. S. Chan, "Noise in homodyne and heterodyne detection," *Opt. Lett.,* vol. 8, pp. 177-179, 1983.
[35] H. Rohde, S. Smolorz, E. Gottwald, and K. Kloppe, "Next generation optical access: 1 Gbit/s for everyone," in *Optical Communication, 2009. ECOC '09. 35th European Conference on*, 2009, pp. 1-3.
[36] H. Rohde, S. Smolorz, J. S. Wey, and E. Gottwald, "Coherent optical access networks," in *Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011 and the National Fiber Optic Engineers Conference*, 2011, pp. 1-3.
[37] S. Smolorz, E. Gottwald, H. Rohde, D. Smith, and A. Poustie, "Demonstration of a coherent UDWDM-PON with real-time processing," in *Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011 and the National Fiber Optic Engineers Conference*, 2011, pp. 1-3.

[38] H. R. Erich Gottwald, Sylvia Smolorz, "Machbarkeit kohärenter Einfaser-UDWM PONs mit 1000 Teilnehmern und 100 km Reichweite," *Photonische Netze,* 2010.
[39] http://www.biccari.com/docs/Fixed_point_tutorial.pdf. (03-07-2012).
[40] R. A. Shafik, S. Rahman, and A. H. M. R. Islam, "On the Extended Relationships Among EVM, BER and SNR as Performance Metrics," in *Electrical and Computer Engineering, 2006. ICECE '06. International Conference on*, 2006, pp. 408-411.
[41] http://www.av.it.pt/osip/. (03-07-2012).
[42] http://www.xilinx.com/support/documentation/user_guides/ug364.pdf. (03-07-2012).
[43] Altera, "Accelerating High-Performance Computing With FPGAs," 2007.
[44] P. Watts, M. Glick, R. Waegemans, Y. Benlachtar, V. Mikhailov, S. Savory, P. Bayvel, and R. I. Killey, "Experimental demonstration of real-time DSP with FPGA-based optical transmitter," in *Transparent Optical Networks, 2008. ICTON 2008. 10th Anniversary International Conference on*, 2008, pp. 202-205.
[45] http://www.xilinx.com/support/documentation/data_sheets/ds150.pdf. (03-07-2012).
[46] http://www.xilinx.com/support/documentation/boards_and_kits/ug533.pdf. (03-07-2012).
[47] http://www.xilinx.com/support/documentation/boards_and_kits/xtp084.pdf. (03-07-2012).
[48] http://www.xilinx.com/tools/feature/csi/coregen.htm. (03-07-2012).
[49] R. H. Katz, *Contemporary Logic Design*: Prentice Hall, 1993.
[50] http://www.xilinx.com/support/documentation/ip_documentation/fifo_generator/v8_3/fifo_generator_ds317.pdf. (03-07-2012).
[51] http://www.xilinx.com/support/documentation/ip_documentation/clk_wiz/v3_2/clk_wiz_ds709.pdf. (03-07-2012).
[52] F. Yangyang, C. Xue, Z. Weiqin, Z. Xian, and Z. Hai, "The Comparison of CMA and LMS Equalization Algorithms in Optical Coherent Receivers," in *Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on*, 2010, pp. 1-4.
[53] J. R. Smith, *Modern Communication Circuits*: McGraw-Hill, 1997.
[54] D. R. Pauluzzi and N. C. Beaulieu, "A comparison of SNR estimation techniques for the AWGN channel," *Communications, IEEE Transactions on,* vol. 48, pp. 1681-1691, 2000.
[55] J. H. Lee, D. K. Jung, C. H. Kim, and Y. C. Chung, "OSNR monitoring technique using polarization-nulling method," *Photonics Technology Letters, IEEE,* vol. 13, pp. 88-90, 2001.
[56] http://www.xilinx.com/support/documentation/ip_documentation/mult_gen_ds255.pdf. (03-07-2012).
[57] http://www.xilinx.com/support/documentation/ip_documentation/cmpy/v5_0/ds793_cmpy.pdf. (03-07-2012).
[58] http://www.xilinx.com/support/documentation/ip_documentation/div_gen_ds530.pdf. (03-07-2012).
[59] http://www.xilinx.com/support/documentation/ip_documentation/cordic_ds249.pdf. (03-07-2012).
[60] J. E. Volder, "The CORDIC Trigonometric Computing Technique," *Electronic Computers, IRE Transactions on,* vol. EC-8, pp. 330-334, 1959.
[61] http://www.xilinx.com/support/documentation/ip_documentation/cordic/v5_0/ds858_cordic.pdf. (03-07-2012).
[62] http://www.xilinx.com/support/documentation/ip_documentation/blk_mem_gen/v6_2/blk_mem_gen_ds512.pdf. (03-07-2012).